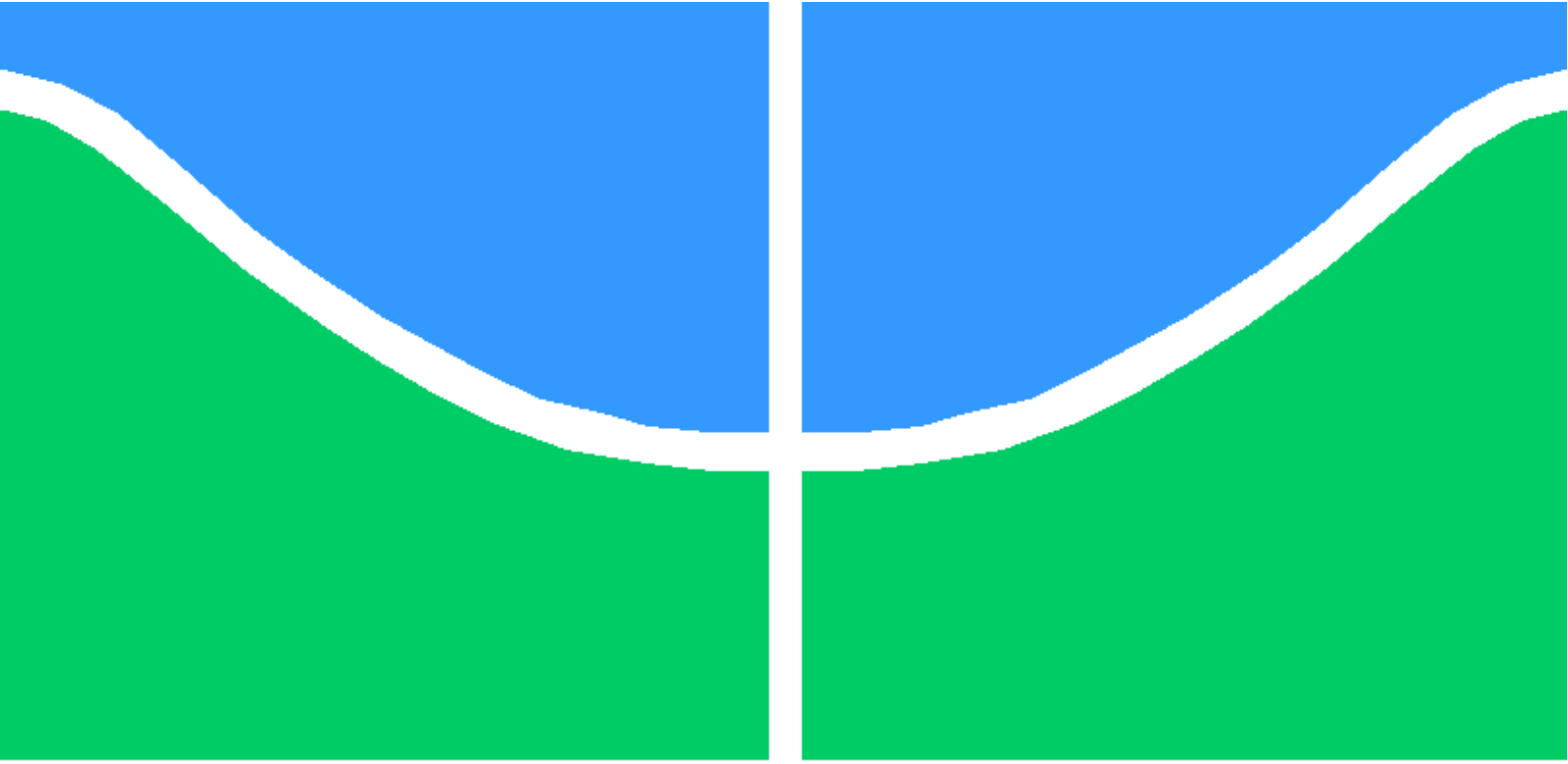


TRABALHO DE CONCLUSÃO DE CURSO

DESENVOLVIMENTO DE UMA PLATAFORMA DE TESTE PARA CONTROLE DE ATITUDE DE HELICÓPTEROS DE PEQUENA ESCALA

Zoé Roberto Magalhães Júnior

Brasília, junho de 2014

UNIVERSIDADE DE BRASÍLIA

FACULDADE GAMA

UNIVERSIDADE DE BRASILIA
Faculdade Gama

TRABALHO DE CONCLUSÃO DE CURSO

DESENVOLVIMENTO DE UMA PLATAFORMA DE TESTE PARA CONTROLE DE ATITUDE DE HELICÓPTEROS DE PEQUENA ESCALA

Zoé Roberto Magalhães Júnior

Relatório submetido ao Departamento de Engenharia Eletrônica como requisito parcial para obtenção do grau de Engenheiro Eletrônico

Banca Examinadora

Prof. Renato Vilela Lopes, FGA/UNB _____
Orientador

Prof. Flávio Henrique Justiniano Ribeiro da Silva, FGA/UnB _____
Examinador

Prof. Marcelino Monteiro de Andrade, FGA/UnB _____
Examinador

Prof. Thiago Felippe Kurudez Cordeiro _____
Examinador

RESUMO

O objetivo deste trabalho é o desenvolvimento de uma plataforma de teste de controladores digitais da atitude de um helicóptero elétrico de pequena escala. A plataforma desenvolvida fixa o helicóptero em uma posição, restringindo os movimentos de translação e permitindo três graus de liberadade de movimentos de rotação, possui um sistema de detecção da atitude executado em computador que fornece em tempo real estimativas da atitude do helicóptero, um sistema de interface envia as medidas dos sensores para o computador e envia os sinais de comando do computador para os motores. O helicóptero usado é o helimodelo elétrico BETL CPX do fabricante Esky®, este helimodelo apresenta módulos de controle e dinâmica semelhantes a um helicóptero convencional. Uma modelagem matemática foi realizada para a resposta do helicóptero aos sinais de controle lateral, longitudinal, pedal e coletivo, a partir de modelos da resposta dos rotores aos sinais de comando e modelos da resposta da atitude do helicóptero a forças geradas pelo movimento das hélice, ao final obteve-se um modelo dedicado para o sistema helimodelo-plataforma que considera a complexidade do BELT CPX, a barra estabilizadora presente em helimodelos e o efeito das forças de apoio e atritos introduzidos pela estrutura física da plataforma. O modelo obtido neste trabalho foi apresentado no artigo Modelagem *Modelagem de um helicóptero elétrico fixo em uma plataforma três graus de liberade* publicado no XIX CREEM em 2012. O sistema de determinação da atitude foi implementado com uma placa de sensores *SEN-10724* fabricada pela *Sparkfun*, que possui acelerômetro, giroscópio e magnetômetro triaxiais. Uma estimativa de atitude é obtida pelo modelo cinemático discreto da propagação da atitude no tempo em função das velocidades angulares medidas pelo giroscópio, e outra estimativa da atitude é obtida pelo método TRIAD aplicado aos vetores de campo gravitacional e magnético medidos pelo acelerômetro e magnetômetro, respectivamente. O Filtro de Kalman Estendido Correlato (FKEC) é usado para fazer a combinação entre as duas estimativas. O FKEC foi implementado em *MATLAB*, e o sistema de interface entre os sensores, computador e motores foi implementada em uma versão baseada em *Arduino* e outra versão baseada em *Raspberry Pi* com *Linux RTOS* embarcado. Testes de desempenho mostraram que as duas versões da interface de aquisição e atuação são capazes de operar a uma taxa de leitura dos sensores e atualização do acionamento dos motores de 50Hz. Simulações realizadas para o modelo matemático mostrou os resultados esperados. E testes realizados com o sistema de determinação da atitude mostrou estimativas de atitude coerentes com os movimentos realizados nos experimentos de teste, e a comparação entre as duas estimativas obtidas sem o FKEC e a estimativa fornecida pelo FKEC permitiu observar o efeito positivo do FKEC. Um modelo em simulação de um sistema de controle da atitude com controladores PIDs discretos e o modelo matemático do helicóptero obtido como planta foi implementado em *Matlab/Simulink* para mostrar um caso típico de utilização da plataforma e demonstrar que controladores simples são capazes de estabilizar o helimodelo na plataforma.

ABSTRACT

The objective of this work is the development of a test platform for digital controls of a small electric helicopter's atitude. The developed platform fixes the helicopter position, blocking the translation and allowing 3 degrees of freedom of rotation, it has a attitude estimation system performs on a computer that provides in real time estimates of the helicopter attitude, a interface system that send measures signals from sensors to computer and command signals form computer to motors. The helicopter used is the Esky®, this model has control models and dynamic similar to to a convencional helicopter. A mathematic modelling was performed for the helicopter response to the control signals lateral, longitudinal, pedal and collective, from models for rotors response to command signals and helicopter's attitude response to forces generated by propeller move, at the end was obtained a dedicated model for the helicopter-platform system that considers the BELT CPX complexity, the fly-bar effect and the effect of support forces and friction introduced by platform's physical structure. The model obtained in this project was presented in the article "Modelagem de um helicóptero elétrico em uma plataforma 3DOF"publish in the XIXCREEM in 2012. The atitude detection system was implemented with SEN-10724 that integrates triaxial accelerometer, gyro, and magnetometer. An atitude estimate is obtained from discrete cinematic model of the time atitude propagation with the angular velocity measure by the gyroscope. Another atitude estimate is obtained from the method TRIAD applied to gravitational and magnetic Earth fields measured by accelerometer and gyro, respectively. The Correlated Extended Kalman Filter (CEKF) combines the two estimates. The CEFK was implemented in Matlab, the interface between computer and sensors and actuators was implemented in version with Arduino and another with Raspberry Pi running embedded RTOS. Simulations for the math model shows the results expected as corrects. And tests for the performed for the attitude detection system shows estimates consistent with movements performed during experiments. A simulation model of a control system with the obtained math model controlled by discrete by PID was implemented in Matlab/Simulink to show the using of the math model in controllers design, the obtained controllers can be tested in the developed platform.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Símbolos Latinos

$\vec{a}_e$	Vetor de medida do acelerômetro com erro	$[m/s^2]$
$\vec{a}$	Vetor de medida do acelerômetro livre de erro	$[m/s^2]$
$\vec{b}$	Vetor de viés do acelerômetro	
$\vec{a}_c$	Vetor de medida do acelerômetro calibrada	$[m/s^2]$
$\vec{A}_x$	Vetor formado pela sequência das medidas do acelerômetro no eixo x	
$\vec{A}_y$	Vetor formado pela sequência das medidas do acelerômetro no eixo y	
$\vec{A}_z$	Vetor formado pela sequência das medidas do acelerômetro no eixo z	
$\vec{e}_b$	Vetor de erro de viés	
$\vec{e}_n$	Vetor de erro aleatório	
e_{pseudo}	Erro de pseudo-observação para manutenção da norma do quatérnio estimado.	
F_x	Matriz de atualização do vetor de estado em função do ultimo vetor de estado	
F_u	Matriz de atualização do vetor de estado em função do ultimo vetor de estado	
$\vec{F}_{MR}$	Vetor força gerado pelo rotor principal	[N]
$\vec{F}_{RC}$	Vetor força gerado pelo rotor de cauda	[N]
$\vec{G}$	Vetor aceleração gravitacional	$[m/s^2]$
$\vec{g}_c$	Vetor de medidas calibradas do giroscópio	[rad/s]
$\vec{g}_e$	Vetor de medidas não calibradas do giroscópio	[rad/s]
H_x	Matriz de comparação entre vetor de estado e vetor de comparação	
HP	Orientação de referência do plano de rotação do rotor principal	
K	Ganho de Kalman	
K_p	Ganho proporcional do controlador PID	
K_i	Ganho de integração do controlador PID	
K_d	Ganho de derivação do controlador PID	
k_x	Fator de correção de escala do eixo x do acelerômetro	
k_y	Fator de correção de escala do eixo y do acelerômetro	
k_z	Fator de correção de escala do eixo z do acelerômetro	
$\vec{l}_{MR}$	Vetor posição do centro de rotação do rotor principal em relação ao centro de articulação da plataforma	[m]

$\vec{l}_{RC}$	Vetor posição do centro de rotação do rotor de cauda em relação ao centro de articulação da plataforma	[m]
$\vec{l}_{CG}$	Vetor posição do centro de gravidade do helicóptero em relação ao centro de articulação da plataforma	[m]
$\vec{M}$	Vetor atração do campo magnético	[T]
$\vec{m}_c$	Vetor de medidas calibradas do magnetômetro	[T]
$\vec{m}_e$	Vetor de medidas não calibradas do magnetômetro	[T]
M_s	Matriz de correção da distorção nas medidas do magnetômetro	
P	Covariância do vetor de estado do filtro de Kaman	
Q_{MR}	Arrasto aerodinâmico	
$\breve{q}$	Quatérnio unitário	
$\vec{q}_x$	Quatérnio estimado pelo Filtro de Kalman Estendido Correlato implementado neste trabalho	
$\vec{q}_y$	Vetor de comparação do Filtro de Kalman Estendido Correlato implementado neste trabalho	
$\vec{s}$	Vetor composto de medidas de sensores	
T_s	Período de amostragem	[s]
T_{MR}	Empuxo gerado pelo rotor principal	[N]
T_{RC}	Empuxo gerado pelo rotor de cauda	[N]
TPP	Plano de rotação do rotor principal	
$\vec{u}$	Vetor de controle do filtro de Kalman	
$\vec{x}$	Vetor de estado do filtro de Kalman	
$\vec{y}$	Vetor de comparação do filtro de Kalman	

Símbolos Gregos

ϕ	ângulo de rolagem	[rad]
θ	ângulo de arfagem	[rad]
ψ	ângulo de guinada	[rad]
$\vec{\omega}$	Vetor velocidade angular	[rad/s]
ω_x	Velocidade angular de rolagem	[rad/s]
ω_y	Velocidade angular de arfagem	[rad/s]
ω_z	Velocidade angular de guinada	[rad/s]
$\Delta\vec{\Theta}$	Vetor rotação	[rad]
α_{MR}	ângulo inicial do segmento de reta entre o centro de articulação da plataforma e o centro do rotor principal	[rad]
α_{RC}	ângulo inicial do segmento de reta entre o centro de articulação da plataforma e o centro do rotor de cauda	[rad]
α_{CG}	ângulo inicial do segmento de reta entre o centro de articulação da plataforma e o centro de gravidade do helimodelo	[rad]

β_{1c}	Ângulo de batimento longitudinal da hélice principal	
β_{1s}	Ângulo de batimento lateral da hélice principal	
$\beta_{fly,1c}$	Ângulo de batimento longitudinal da barra estabilizadora	[rad]
$\beta_{fly,1s}$	Ângulo de batimento lateral da barra estabilizadora	[rad]
ζ_{mr}	Constante de amortecimento do rotor principal	
ζ_{fly}	Constante de amortecimento do rotor principal	
τ_{MR}	Torque gerado pelo rotor principal	[Nm]
τ_{RC}	Torque gerado pelo rotor de cauda	[Nm]
τ_d	Torque de reação ao movimento do rotor principal	[Nm]
τ_G	Torque gerada pela gravidade	[Nm]
τ_{at}	Torque de atrito introduzido pela plataforma	[Nm]
τ_{eg}	Torque de efeito giroscópico	[Nm]
$\vec{\eta}_u$	Covariância do vetor de controle	
$\vec{\eta}_s$	Covariância do vetor de medidas de sensores	

Subscritos

x	eixo x
y	eixo y
z	eixo z

Sobrescritos

$\overset{O}{0}$	Representação no sistema de coordenadas O_0
$\overset{O}{1}$	Representação no sistema de coordenadas O_1
$\overset{O}{2}$	Representação no sistema de coordenadas O_2
$\overset{O}{3}$	Representação no sistema de coordenadas O_3

Siglas

VANT	Veículo aéreo não tripulado
FKEC	Filtro de Kalman Estendido Correlato
SNI	Sistema de navegação inercial
UMI	Unidade de medida inercial

1 INTRODUÇÃO

"Alice: Would you tell me, please, which way I ought to go from here?
The Cheshire Cat: That depends a good deal on where you want to get to.
Alice: I don't much care where.
The Cheshire Cat: Then it doesn't much matter which way you go.
Alice: ...So long as I get somewhere.
The Cheshire Cat: Oh, you're sure to do that, if only you walk long enough "

- Lewis Carroll, Alice in Wonderland

1.1 CONTEXTUALIZAÇÃO

Os helicópteros não tripulados são os veículos mais flexíveis dentro de uma grande variedade de VANTs (Veículos Aéreos Não Tripulados), pois apresentam a capacidade de realizar pouso e decolagem vertical, alta manobrabilidade que permite executar manobras agressivas e trabalhos em ambientes de espaço limitado, e característica inerente de vôo pairado que confere a vantagem de realizar observações eficientes em diferentes ângulos [4]. Essas características em conjunto com o desenvolvimento contínuo das tecnologias utilizadas nos veículos aéreos não tripulados possibilitaram a utilização do helicóptero não tripulado em muitas aplicações civis e militares como por exemplo, monitoramento para prevenção de desastres ambientais, inspeção de linhas de transmissão de energia elétrica, operações de busca e salvamento, vigilância e monitoramento de regiões urbanas e fronteira [5], [6].

Assim como todo veículo que ocupa o espaço aéreo e por executar tarefas que exigem confiabilidade das ferramentas utilizadas, os VANTs precisam ser rigorosamente testados antes de serem lançados como produtos. Por este motivo, a pesquisa e desenvolvimento de metodologia e ferramentas de testes é uma das principais áreas relacionadas com projeto de VANTs.

Na literatura, é possível encontrar trabalhos de vários grupos de pesquisa que desenvolveram plataformas de avaliação de helicópteros. E como neste trabalhos, existe a preocupação em minimizar os feitos da variação da dinâmica do veículo causado pela plataforma e um processo cuidadoso de modelagem matemática do helicóptero e/ou plataforma. O laboratório de instrumentação e controle do Instituto de Tecnologia Federal Suíço de Zurique desenvolveu uma plataforma com seis graus de liberdade que permite ao helimodelo a mesma mobilidade de vôo livre [7] apud [8]. Já em [9] projetou-se uma plataforma com três graus de liberdade que permite que o movimento do veículo na plataforma seja o mais próximo possível de um helimodelo em um vôo pairado. Em [10] os autores realizam o projeto e avaliação de um controle e estabilização do vôo de um helicóptero do tipo quadri-rotor conectado a uma plataforma com três graus de

liberdade.

No Brasíl, a Universidade de Brasília é uma das pioneiras no desenvolvimento de plataformas de testes. Em [11], foi desenvolvida uma plataforma com três graus de liberdade para o estudo do movimento angular de helicópteros. Os autores usam um pequeno helicóptero elétrico, bastante simples e que não possui todos os comandos de um helicóptero real, preso a uma junta cardan que permite movimento nos três ângulos de rotação. O modelo utilizado, por sua fragilidade, pequena capacidade de carga e dinâmica distante de um helicóptero real, foi posteriormente substituído por um helicóptero a combustão [12].

1.2 OBJETIVOS DO PROJETO

O objetivo deste trabalho é apresentar soluções que contribuam para o desenvolvimento de uma plataforma de teste para sistemas de controle da atitude de um helicóptero elétrico com pequeno porte que permita testes de controladores digitais dos movimentos de rotação da aeronave. Entende-se como atitude o comportamento da orientação do helicóptero representado por variáveis que descrevam os movimentos de rotação do helicóptero em função do tempo, por exemplo ângulos e velocidades angulares.

Os objetivos específicos deste trabalho são

- Projetar e fabricar a estrutura física de fixação do helicóptero em um ponto fixo com 3 graus de liberdade dos movimentos de rotação
- Obter um modelo matemático da resposta da atitude do helicóptero fixo a plataforma aos sinais de controle
- Desenvolver o sistema de estimação da atitude que forneça em tempo real estimativas da atitude do helicóptero a serem usadas como sinal de realimentação do sistema de controle em malha fechada.
- Implementar em simulação um sistema de controle para o modelo do helicóptero obtido.

1.3 JUSTIFICATIVA

O mercado de VANTs teve um notável crescimento e hoje inclui veículos de vários tipos, tamanhos e capacidades operacionais [13]. No entanto, na última década o interesse tem sido maior em helicóptero de pequena escala (cerca de 150cm de comprimento) para o desenvolvimento e experimentação devido principalmente ao seu baixo custo [4]. Porém, por outro lado, os helicópteros apresentam dinâmica complexa, não-linear, inerentemente instável e com modos fortemente acoplados o que torna o desenvolvimento de sistemas autônomos de navegação um procedimento difícil e perigoso.

Uma vez que é difícil o desenvolvimento desses sistemas de controle, a realização de experimentos práticos oferece freqüentemente riscos de acidentes. Esse risco inerente, obriga a realização de vôos em ambientes externos seguros (ao ar livre), com tempo de operação limitado devido às condições meteorológicas. Além disso, é recomendável, em todos os testes, a utilização de um piloto experiente para

tentar contornar eventuais problemas causados por alguma falha em vôo, que são inevitáveis no início do desenvolvimento [9]. Esses fatores causam, além da elevação do custo do projeto, atraso no tempo de desenvolvimento do projeto devido principalmente ao tempo que o veículo e subsistemas permanecem em manutenção após ocorrência de falha do sistema de controle durante teste.

Por essas razões, o desenvolvimento de sistemas de controle para navegação autônoma começa com inúmeros testes em ambientes de simulação. Nestes ambientes, os controladores são avaliados quanto à sua capacidade de controlar eficientemente um modelo matemático do helicóptero [4]. No entanto, o procedimento de simulação têm alguns inconvenientes. Primeiro, o ambiente de simulação não consegue reproduzir, em detalhes, todos os possíveis distúrbios do sistema. Em segundo, o controlador pode apresentar um desempenho satisfatório em simulação mas ser ineficiente quando aplicado ao veículo real. Desta forma, as simulações não conseguem eliminar totalmente os riscos de acidentes nos testes iniciais. Assim é desejável testar o controlador no veículo, mas em um ambiente seguro, sem ter perigo de destruir o equipamento ou causar danos às pessoas que monitoram o vôo.

Uma alternativa é o desenvolvimento de plataformas onde os helimodelos ficam presos a pontos de testes, limitando seus movimentos a um determinado espaço e configuração de trabalho. Essas plataformas geralmente são utilizadas como um método inicial para analisar o comportamento dinâmico desses veículos, ou de algum movimento em específico (arfagem, guinada, etc). Por serem também estruturas rígidas, elas podem ser instrumentadas com sensores de movimento para medir a posição e orientação do helicóptero, poupando de início necessidade de sensores inerciais embarcados, bem como projeto do sistema de localização. Adicionalmente, podem ser utilizadas como sistema de calibração destes sensores inerciais. Essas plataformas podem servir ainda para estudos na área de identificação de sistemas e de metodologias de controle [8].

Devido à importância de plataformas de testes para desenvolvimento de VANTs, justifica-se este trabalho cujo o objetivo é desenvolver uma plataforma de teste para sistema de controle de helicóptero helicóptero em pequena escala para permitir, de forma rápida e segura, o desenvolvimento de controladores para estabilização da aeronave. Este trabalho apresenta soluções para as funcionalidades essenciais da plataforma de teste proposta, e permitirá a realização de novas pesquisas na Universidade de Brasília-Faculdade Gama, servindo como ferramenta de implementação e teste de controladores de atitude de veículos aéreos elétricos de pequeno porte.

1.4 ESTRUTURA

O primeiro capítulo deste documento, apresenta a motivação do trabalho e seus objetivos. O Capítulo 2 apresenta uma revisão bibliográfica dos conhecimentos específicos, não amplamente conhecidos, fundamentais para o desenvolvimento e compreensão do trabalho.

O capítulo 3 descreve o desenvolvimento dos subsistemas: construção da estrutura física, levantamento do modelo matemático da atitude do helicóptero, implementação da interface em tempo real de aquisição de dados proveniente de sensores e comunicação entre os subsistemas da plataforma, desenvolvimento do sistema de detecção da atitude com Filtro de Kalman Estendido Correlato.

O capítulo 4 apresenta os resultados empíricos do modelo matemático da resposta do helicóptero fixo à plataforma aos sinais de comando lateral, longitudinal, coletivo e pedal, um modelo em simulação do sistema de controle para o modelo matemático obtido utilizando controladores PIDs discretos, teste de desempenho do sistema de comunicação entre sensores, motores e computador e testes de qualidade do sistema do sistema de detecção da atitude.

2 REVISÃO BIBLIOGRÁFICA

"Não sei explicar. Só sei que em certos momentos a gente muda de estado e começa a ver as maravilhosas coisas que estão em redor de nós "

-Monteiro Lobato, Reinações de Narizinho

2.1 INTRODUÇÃO

Os conceitos utilizados durante esse trabalho não são amplamente conhecidos. Antes de apresentar o trabalho feito para desenvolvimento da plataforma, será apresentada uma pequena revisão dos principais conceitos específicos utilizados durante esse trabalho.

2.2 DINÂMICA DE UM HELICÓPTERO

Em um helicóptero convencional, a força de sustentação é gerada pelo rotor principal que é responsável ainda por gerar as forças de tração que proporcionam os movimentos de translação. Nestes helicópteros, o piloto possui o comando coletivo U_{col} para controlar a magnitude do empuxo gerado pelo rotor principal, e os comandos cíclico lateral U_{lat} e cíclico longitudinal U_{lon} para controlar a direção. A atuação desses comandos, ilustrada na figura 2.1 é feita por meio de uma bailarina (*swash plate*, termo em inglês), que é uma estrutura composta por um prato fixo ao corpo do helicóptero e um prato que gira acoplado ao eixo do rotor principal [1].

O comando coletivo, por meio dos atuadores, altera o ângulo de ataque das hélices de um valor constante em todo o percurso da hélice no plano de rotação. Dessa forma, o comando coletivo controla a

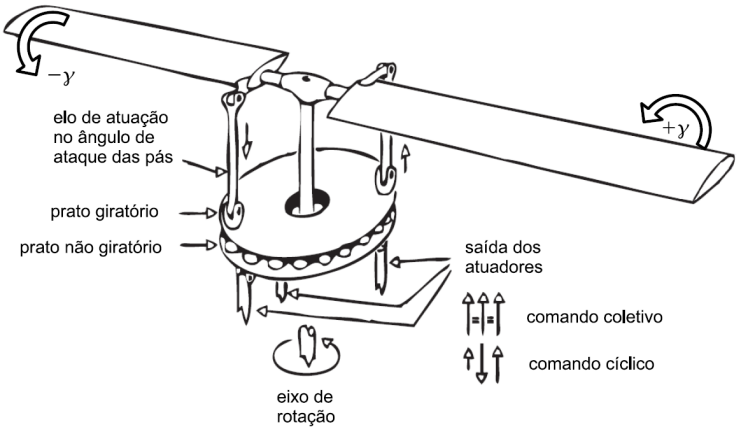


Figura 2.1: Controle do rotor principal [1].

magnitude da força gerada pelo rotor principal. Já os comandos cíclicos inclinam os pratos da bailarina provocando uma variação no ângulo de ataque das hélices de um valor que é função da posição da hélice no plano de rotação, o que altera a direção da força de sustentação.

Por conservação de momento angular, a quantidade de movimento angular transferida para a hélice do rotor principal é igual a quantidade de movimento transferida da hélice para a fuselagem do helicóptero. Assim, o movimento do rotor principal causa um torque na fuselagem que induz uma rotação no sentido contrário ao sentido de rotação das pás da hélice. O rotor de cauda dos helicópteros possui a função de gerar um torque que se oponha ao torque de reação ao movimento do rotor principal, mas como efeito, a força do rotor de cauda gera um torque na fuselagem que faz com que o helicóptero tenda a apresentar um movimento de rolagem. Numa situação de vôo pairado, o equilíbrio é conseguido pela ação do rotor principal. Por fim, o rotor de cauda tem a função ainda de atuar no ângulo de guinada da aeronave, pela variação do ângulo de ataque de suas pás. O comando utilizado pelo piloto para controlar o ângulo de ataque do rotor de cauda é chamado comando de pedal (U_{ped}).

As forças aerodinâmicas produzidas pelo rotor principal para gerarem a sustentação e as acelerações necessárias para os movimentos são grandes quando comparadas à massa das pás do rotor, o que provoca uma tendência natural das pás responderem muito rapidamente aos comandos cíclicos, o que dificulta o controle do helimodelo pelo piloto humano[14]. Para superar essa dificuldade os helimodelos empregam a barra estabilizadora conhecida como misturadora de Bell-Hiller ou simplesmente *fly-bar*. Esta barra funciona como um segundo par de hélices, instaladas a 90° das hélices do rotor principal, e não geram sustentação ou arrasto, pois não estão submetidas aos comando de coletivo, apenas aos comandos cíclicos. Em outras palavras, a função da barra estabilizadora é deixar o sistema mais lento, dando uma impressão de estabilidade ao piloto. Dessa forma, para obter um modelo matemático adequado ao helimodelo é fundamental realizar a modelagem dos efeitos da barra estabilizadora.

Observa-se na dinâmica do helicóptero diversos acoplamentos entre os diferentes movimentos da aeronave e, devido ao efeito giroscópico, acoplamentos entre os comandos cíclicos que alteram a orientação do plano de rotação do rotor principal. Esses fatores e a quantidade de variáveis de entrada e saída envolvidas tornam a dinâmica de um helicóptero extremamente complexa e, por isso, buscam-se modelos simplificados que representam satisfatoriamente o comportamento do veículo.

2.3 SISTEMAS DE DETERMINAÇÃO DA ATITUDE

Sistemas de navegação inercial (SNI) são sistemas capazes de fornecer a orientação, a posição e a velocidade de acordo com uma referência. Os SNI são formados por unidade de medida inercial (UMI), computador de navegação e interface [2]. Chama-se de sistemas de determinação da atitude da atitude (SDA) os SNIs que fornecem apenas a atitude do corpo representada por variáveis o estado de movimento rotacional do corpo, por exemplo, ângulos de Euler e velocidades angulares.

A unidade de medida inercial é o conjunto de sensores inerciais associados com seus mecanismos de controle e calibração que fornecem medidas de aceleração e rotação em relação a um mesmo sistema de coordenadas. A aceleração é obtida com o acelerômetro, esse sensor inercial mede a força específica, definida

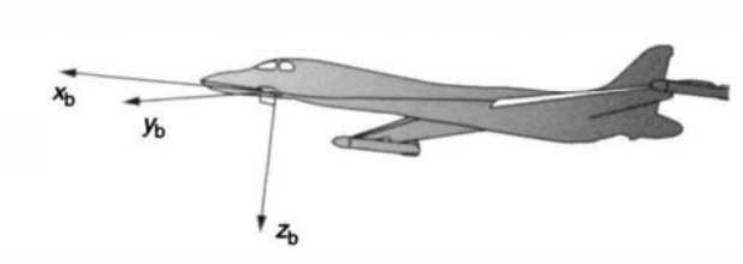


Figura 2.2: Eixos coordenados do Sistema de referência *Body b-frame* [2].

como força por unidade de massa, aplicada ao sensor desprezando a aceleração gravitacional, de forma que um acelerômetro instalado em um ponto em repouso em relação à Terra mede a força específica normal gerada pelos apoios de sustentação. Sendo a força específica normal o oposto da aceleração gravitacional, pode-se dizer que o acelerômetro sem movimentos de translação mede a aceleração gravitacional com um ganho de sinal negativo. O movimento de rotação é medido com o giroscópio, esse sensor inercial mede a velocidade angular do corpo em relação a um eixo de medida. A associação de três sensores do mesmo tipo com eixos de medidas independentes, idealmente ortogonais, é chamada de triaxial. Uma unidade de medida inercial completa possui um acelerômetro triaxial e um giroscópio triaxial, ambos com os mesmos eixos de medida. Os mecanismos de medidas dos sensores inerciais são apresentados com mais detalhes na Seção 2.4.

O computador de navegação é responsável por realizar correções nas medidas obtidas da unidade de medida inercial e executar o algoritmo estimação da orientação, velocidade e posição do corpo em relação ao sistema de coordenadas inerciais do SNI.

As referências a seguir são utilizadas em sistemas de estimação de atitude [2].

Sistema de referência Body b-frame: Fixo em relação ao corpo alvo, normalmente com origem no centro de massa ou centro geométrico do corpo, de forma que a posição e orientação do corpo no *b-frame* não muda. A Figura 2.2 apresenta uma ilustração do sistema de coordenada *b-frame*.

Sistema de referência Navegação n-frame: Origem em um ponto de uma base tomada como referência dos movimentos medidos. Orientação fixa em relação à Terra e coincidente com *b-frame* quando o corpo encontra-se na orientação definida como referência.

2.3.1 Estimativa da atitude

Adotando o *n-frame* como o sistema de referência do sistema de detecção da atitude e *b-frame* como sistema de referência dos sensores de medida inercial, a atitude pode ser calculada pela integração no tempo das medidas de velocidade angular obtidas do giroscópio, mas essas medidas possuem erros que são acumulados pela integração, fazendo com que a estimativa apresente um erro crescente no tempo.

Uma alternativa é utilizar dois ou mais vetores não paralelos, cada um medido no *n-frame* e *b-frame* e utilizar algoritmos que determinam a atitude por combinação matemática destes vetores. No entanto, vibrações, interferências ou variações bruscas introduzirão erros na atitude estimada.

Um melhor resultado é obtido pela combinação dos dois métodos [15]. Por exemplo, a combinação com Filtro de Kalman da atitude calculada com as medidas de velocidades angulares e da atitude estimada pelos vetores de aceleração gravitacional e campo magnético terrestre nos sistemas *n-frame* e *b-frame* combinados pelo algoritmo TRIAD, feita em [16].

2.4 SENSORES MEMS

Micro-machined electromechanical systems (MEMS) são sistemas mecânicos fabricados na escala de circuitos integrados, o que permite que sejam embarcados em CI's para composição de sensores. Sensores MEMS apresentam baixo custo e escala, por estes motivos estes dispositivos estão entre os principais desenvolvimentos de sensores dos últimos 25 anos [2], [17]. Nesta seção serão apresentados os três sensores usados neste trabalho: acelerômetro, giroscópio e magnetômetro.

Acelerômetros medem a força específica externa (força por unidade de massa) que age sobre uma massa de prova interna conectada por uma suspensão, formando um sistema massa-mola. Quando uma força de campo ou força de aceleração é aplicada ao acelerômetro, a massa de prova sofre um deslocamento proporcional à força específica. Este deslocamento interfere no estado elétrico de algum componente do sensor, fazendo com que a saída varie em função da força específica aplicada ao sensor. A saída de um acelerômetro é composta por dois termos aditivos, o primeiro proporcional à atração gravitacional e o segundo proporcional à aceleração linear do sensor [2]. Um acelerômetro em repouso apresenta saída proporcional à força de atração gravitacional. Em sistemas de estimativa da atitude de corpos com posição fixa (aceleração linear nula), acelerômetros podem ser utilizados para medir o vetor de atração gravitacional terrestre, este vetor pode ser utilizado como entrada de algoritmos de estimação da atitude como o algoritmo TRIAD [16]. Nessas aplicações, movimentos de translação ou vibrações atuarão como fontes de erro.

Giroscópios do tipo MEMS são compostos por uma massa em movimento harmônico simples em torno de um eixo de vibração. A rotação do encapsulamento em torno de um eixo perpendicular ao eixo de vibração, por efeito de Coriolis, faz com que a massa vibrante apresente vibrações perpendiculares ao eixo de vibração com magnitude proporcional à velocidade angular do encapsulamento. A componente de vibração proporcional à velocidade angular pode ser medida por sensores capacitivos [16]. Em sistemas de estimativa da atitude a velocidade angular medida pelo giroscópio pode ser integrada no tempo para a atualização temporal da atitude, mas erros de medida não corrigidos serão propagados e acumulados no tempo.

Magnetômetros típicos são baseados no efeito Hall [18]. Em sistemas de estimativa da atitude, magnetômetros são usados para medir o vetor campo magnético terrestre que pode ser utilizado em algoritmos de estimação da atitude, como o algoritmo TRIAD. Nessas aplicações, a presença de outros campos magnéticos ou materiais ferromagnéticos atuará como fonte de erro.

Assim como todo sensor, os sensores MEMS apresentam erros em suas medidas. As principais fontes de erro são variação do fator de escala, viés, desalinhamento e ruído. O fator de escala é a razão entre a entrada e a saída do sensor. Normalmente-se assume-se que o fator de escala é constante e igual para todos os eixos, mas na prática o fator de escala pode variar em função da entrada ou variar no tempo e diferir

entre os eixos de medida. O viés é um erro aditivo constante que desloca a saída em uma direção, de forma que para uma entrada nula a saída será igual ao viés. O desalinhamento é o erro obtido ao assumir que sensores triaxiais apresentam eixos de medida perfeitamente perpendiculares enquanto os ângulos reais são diferentes de $90°$.

Um modelo simplificado que relaciona o vetor de saída do sensor $M_{out} = [M_{out}^x \ M_{out}^y \ Mout^z]$ com a entrada sem erro $M = [M^x \ M^y \ M^z]$ é obtido considerando que a saída possui um vetor de erro aditivo determinístico constante e_b, composto pelo viés, e um vetor de erro aditivo aleatório, composto pela variação do fator de escala no tempo e pelo ruído de média nula, e que a relação da saída M_{out} com a entrada M é dada por uma matriz A de transformação tridimensional [2].

$$M_{out} = AM + \vec{e}_b + \vec{e}_n \tag{2.1}$$

A matriz A e o vetor $\vec{e}_b$ são corrigidos em calibração e o ruído aleatório $\vec{e}_n$ pode ter suas propriedades estatísticas estimadas e utilizadas em projeto de filtros que reduzam seu efeito [19].

2.4.1 Representação da atitude

As principais formas de representação da atitude de um corpo são ângulos de Euler, quatérnios unitários e matriz de rotação.

Os ângulos de Euler expressam a orientação do corpo por três ângulos de rotação: rolagem θ, arfagem ϕ e guinada ψ em torno dos eixos x, y e z, respectivamente, de *b-frame*. A atitude representada depende da ordem das rotações e ângulos de Euler diferentes podem apresentar a mesma orientação, e se a representação for limitada em um único ciclo (0 rad a 2π rad) existirá singularidade na representação de movimentos que passam pelos extremos 0 rad e 2π rad [20]. A figura 2.3 apresenta os ângulos de Euler.

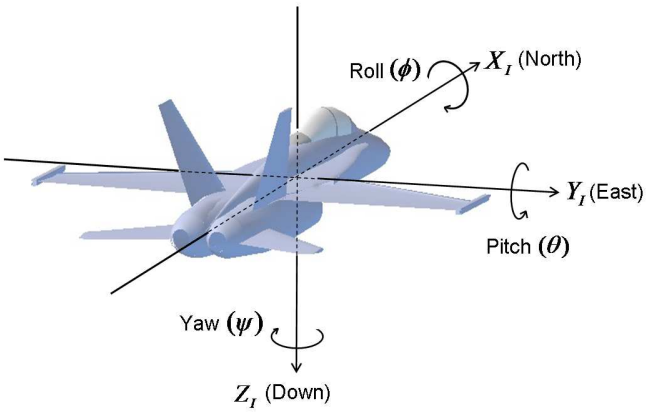


Figura 2.3: Ângulos de Euler [3].

A matriz de cossenos diretores, ou matriz de rotação, é uma matriz quadrada 3x3 que relaciona a representação de um vetor em um sistema de coordenadas O_a com sua representação em um outro sistema de coordenadas O_b rotacionado em relação ao sistema O_a.

Os quatérnios são números hipercomplexos com três elementos imaginários, os quatérnios unitários com parte real positiva podem ser utilizados para representar orientações de forma biunívoca e sem singularidades, em outras palavras, cada quatérnio unitário representa apenas uma orientação e dois quatérnios unitários com parte real positiva não representam a mesma orientação e qualquer movimento contínuo terá representação contínua.

$$\breve{q} = q_0 + q_1\hat{i} + q_2\hat{j} + q_3\hat{k} \equiv \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \tag{2.2}$$

A representação em ângulo de Euler é utilizada para visualização e compreensão da atitude estimada, por ser a representação mais intuitiva. A representam em matriz de rotação é utilizada por algoritmos de estimação baseados em vetores observados em cada orientação como por exemplo os métodos TRIAD, SVD e FAST. Os quatérnios são usados em modelos de propagação da atitude no tempo, pois não possuem singulares como os ângulos de Euler e apresentam menor esforço computacional do que a propagação com matrizes de rotação, pois possuem menos termos a serem calculados [20].

2.4.2 Propagação da atitude em quatérnios

Em [21], a propagação no tempo da atitude em quatérnios em função do vetor do velocidade angular $\vec{\omega}$ é dada pela equação 2.3a.

$$\dot{\breve{q}}(t) = \frac{1}{2}\begin{bmatrix} \vec{\omega} \\ 0 \end{bmatrix}\breve{q}(t) \tag{2.3a}$$

$$\vec{\omega} = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{2.3b}$$

em $\otimes$ indica o produto de quatérnio, e pela definição de produto de quatérnios a equação 2.3a pode ser escrita como na equação 2.4a.

$$\dot{\breve{q}}(t) = \frac{1}{2}W\breve{q}(t) \tag{2.4a}$$

$$W = \Omega(\vec{\omega}) = \begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix} \tag{2.4b}$$

Se o vetor velocidade angular $\vec{\omega}$ for constante no intervalo de tempo entre t e $t + \Delta T$ e se $\Delta\theta$ for suficiente pequeno para que se comporte como um vetor, a solução da equação 2.3a, de acordo com [21], pode ser aproximada pela equação 2.6a.

$$\Delta\vec{\Theta} = \begin{bmatrix} \Delta\theta \\ \Delta\phi \\ \Delta\psi \end{bmatrix} = \int_{t}^{t+\Delta T} \vec{\omega}\,\mathrm{d}t \tag{2.5}$$

$$\breve{q}(t+\Delta t) \approx M(\Delta\vec{\Theta})\breve{q}(t) \tag{2.6a}$$

$$M(\Delta\vec{\Theta}) = \cos\left(\frac{\left|\Delta\vec{\Theta}\right|}{2}\right)\mathrm{I}_{4x4} + \frac{\sin\left(\frac{|\Delta\vec{\Theta}|}{2}\right)}{\left|\Delta\vec{\Theta}\right|}\Omega\left(\Delta\vec{\Theta}\right) \tag{2.6b}$$

$$\Omega\left(\Delta\vec{\Theta}\right) = \begin{bmatrix} 0 & \Delta\psi & -\Delta\theta & \Delta\phi \\ -\Delta\psi & 0 & \Delta\phi & \Delta\theta \\ \Delta\theta & -\Delta\phi & 0 & \Delta\psi \\ -\Delta\phi & -\Delta\theta & -\Delta\psi & 0 \end{bmatrix} \tag{2.6c}$$

Fazendo $q\,[k] = q(k\Delta t)$ e assumindo que o vetor velocidade angular é constante entre os instantes $k\Delta t$ e $(k+1)\Delta t$ para qualquer k, então o $\Delta\vec{\Theta}\,[k]$ pode ser calculado pela equação 2.7, e a equação discreta da propagação da atitude em quatérnios em função das amostras de velocidade angular pode ser escrita como na equação 2.8a

$$\Delta\vec{\theta} = \vec{\omega}\Delta t \tag{2.7}$$

$$\breve{q}\,[k] = \left[\cos\left(\frac{1}{2}\left|\vec{\omega}\,[k]\right|\Delta t\right)\mathrm{I}_{4x4} + \frac{\sin\left(\frac{1}{2}\left|\vec{\omega}\,[k]\right|\Delta t\right)}{\left|\vec{\omega}\,[k]\right|\Delta t}\Omega\left(\vec{\Delta}\Theta\,[k]\right)\right]\breve{q}\,[k-1] \tag{2.8a}$$

$$\Omega\left(\Delta\vec{\Theta}\right) = W\Delta t \tag{2.8b}$$

2.5 MÉTODO TRIAD

O Algoritmo TRIAD foi desenvolvido por Harold D. Black em 1964 para estimar a atitude de um corpo a partir da observação de dois vetores não colineares no sistema de coordenadas de referência e em um sistema de coordenadas fixo em relação corpo, sendo que os dois sistemas de coordenadas coincidem quando o corpo está na orientação adotada como origem dos movimentos de rotação [22].

Adotando *n-frame* como o sistema de coordenadas de referência e *b-frame* como o sistema de coordenadas fixo em relação ao corpo e sendo $\vec{M}$ o vetor de atração do campo magnético terrestre e $\vec{G}$ o vetor de atração do campo gravitacional terrestre na posição do corpo, de acordo com o algoritmo apresentado em [22] a matriz de atitude do corpo pode ser estimada pela equação 2.9a.

$$\mathrm{DCM} = \begin{bmatrix} \vec{i}_b & \vec{j}_b & \vec{k}_b \end{bmatrix}\begin{bmatrix} \vec{i}_n & \vec{j}_n & \vec{k}_n \end{bmatrix}^T \tag{2.9a}$$

$$\vec{i} = \frac{\vec{G} + \vec{M}}{\left|\vec{G} + \vec{M}\right|} \tag{2.9b}$$

$$\vec{j} = \frac{\vec{i} \times \left(\vec{G} - \vec{M}\right)}{\left|\vec{i} \times \left(\vec{G} - \vec{M}\right)\right|} \tag{2.9c}$$

$$\vec{k} = \vec{i} \times \vec{j} \tag{2.9d}$$

em que na Equação 2.9a os sub-índices b e n indicam que o vetor está representado em *b-frame* e *n-frame*, respectivamente.

2.6 FUSÃO SENSORIAL POR FILTRO DE KALMAN

O Filtro de Kalman foi desenvolvido por R. E. Kalman e publicado em 1960 como ferramenta para problemas práticos de controle de natureza estatística como predição de sinais aleatórios, separação de sinais aleatórios de ruído aleatório e predição de sinais de forma conhecida em presença de ruído, e é um método eficiente e versátil para estimação de estados de sistema a partir de sensores ruidosos [19].

As variáveis estimadas são representadas em um vetor de estado, em que cada elemento do vetor de estado corresponde a uma variável estimada pelo filtro. O processo de estimação é realizado em duas etapas: predição e correção. Durante a predição é realizada a propagação no tempo do vetor de estado $\vec{x}$ e da matriz de covariância P do vetor estimado, a propagação temporal é representada por um modelo matemático que relaciona o estado atual com o último estado estimado e um vetor de controle formado com variáveis de atualização calculadas a partir de medidas de sensores obtidas de forma independente do estado estimado. Durante a correção o vetor de estado e as estimativas de incerteza são corrigidos por um modelo que compara o vetor de estado, formado por medidas preditas, com um vetor de comparação, formado por medidas realizadas.

A formulação do Filtro de Kalman considera que o vetor de estado $\vec{x}$ possa ser predito em função do vetor de estado estimado em instante anterior e do vetor de controle por um modelo na forma da equação 2.10a. E considera que o vetor de comparação formado por medidas realizadas possa ser comparado com o vetor de estado (medidas preditas) por um modelo na forma da equação 2.10b.

$$\vec{x}\,[k] = F_x \vec{x}[k-1] + F_u \vec{u}\,[k] + \vec{w}_p\,[k] \tag{2.10a}$$

$$\vec{y}\,[k] = H_x \vec{x}\,[k] + \vec{w}_c\,[k] \tag{2.10b}$$

em que $\vec{w}_p$ e e $\vec{w}_c$ são os vetores de ruído de predição e de correção, respectivamente, assumidos como não correlatos. A equação 2.10b apresenta apenas a relação do vetor de comparação com o vetor de estado. O vetor de comparação $\vec{y}$ é calculado em função do vetor de medidas $\vec{s}$ e do vetor de controle $\vec{u}$.

$$\vec{y}\,[k] = g(\vec{s}, \vec{u}) \tag{2.11}$$

Para aplicações não lineares, um modelo linear na forma da Equação 2.10a pode ser obtido por linearização. O Filtro de Kalman Estendido(FKE) foi introduzido por Stanley F. Schmidt [19], em sua formulação as matrizes F_x, F_u e H_x correspondem às jacobianas das funções $f(\vec{x}, \vec{u})$ e $h(\vec{x})$.

A premissa de que os ruídos de predição e de correção são não-correlatos não é satisfeita em todas as aplicações. A formulação do filtro de Kalman que considera a covariância entre os ruídos de predição e correção é chamada de Filtro de Kalman Correlato (FKC). O FKC modela a covariância entre os ruídos em três matrizes R, S e Q, apresentadas na equação 2.12.

$$E\left[\begin{pmatrix} \vec{w}_p \\ \vec{w}_c \end{pmatrix}\begin{pmatrix} \vec{w}_p^T & \vec{w}_c^T \end{pmatrix}\right]_{2n\times 2n} = \begin{bmatrix} Q_{n\times m} & S_{n\times m} \\ S_{n\times m} & R_{n\times m} \end{bmatrix} \cdot \delta(k,j) \tag{2.12}$$

em que n é o comprimento do vetor de estados, m é o comprimento do vetor de comparação e $\delta(k,j)$ é delta de Kronecker.

2.6.1 Filtro de Kalman Estendido Correlato

O Filtro de Kalman Estendido Correlato (FKEC) é uma formulação utilizada para processos não lineares em que assume-se que a covariância entre os ruídos das etapas de predição e correção é não-nula, o que ocorre quando a correção tem correlação com a predição anterior, ou, a predição tem correlação com a correção anterior. O projeto do FKEC começa com a definição dos vetores de estado, de controle de comparação e o levantamento de um modelo para o vetor de estado em função do último vetor de estado estimado e do vetor de controle, e um modelo da relação entre o vetor de comparação em função do vetor de estado. As equações 2.13a e 2.13b apresentam o formato da relação entre os vetores de estado, de controle e de comparação.

$$\vec{x}\,[k] = f(\vec{x}[k-1], \vec{u}\,[k]) \tag{2.13a}$$

$$\vec{y}\,[k] = h(\vec{x}\,[k]) \tag{2.13b}$$

As matrizes de atualização F_x e F_u corresponde as jacabianas de $f\,(\vec{x}, \vec{u})$ em relação a $\vec{x}$ e $\vec{u}$ respectivamente, avaliadas em $\vec{x} = \vec{q}_x[k-1]$ e $\vec{u} = \vec{u}\,[k]$. E a matriz de comparação H_x corresponde a jacaboana de $h(\vec{x})$ avaliada em $\vec{x} = \vec{q}_x\,[k]$. A equações 2.14, 2.15 e 2.16 enfatizam a definição dessas matrizes no FKEC.

$$F_x\,[k] = \left.\frac{\partial f(\vec{x}, \vec{u})}{\partial \vec{x}}\right|_{\vec{x}=\vec{x}[k-1], \vec{u}=\vec{u}[k]} \tag{2.14}$$

$$F_u\,[k] = \left.\frac{\partial f(\vec{x}, \vec{u})}{\partial \vec{u}}\right|_{\vec{x}=\vec{x}[k-1], \vec{u}=\vec{u}[k]} \tag{2.15}$$

$$H_x\left[k\right] = \left.\frac{\partial h(\vec{x})}{\partial \vec{x}}\right|_{\vec{x}=\vec{x}[k]} \tag{2.16}$$

As jacobianas do vetor de comparação em função do vetor de medidas $\vec{s}$ e $\vec{u}$, respectivamente, são utilizadas no FKEC e precisam ser calculadas a cada nova estimativa. As equações 2.17a e 2.17b enfatizam as definições das matrizes G_s e G_u. E uma das fontes de correlação entre os ruídos de correção e predição pode ser a dependência do vetor de comparação com o vetor de estado.

$$G_s\left[k\right] = \left.\frac{\partial g(\vec{s},\vec{u})}{\partial \vec{s}}\right|_{\vec{s}=s[k],\vec{u}=u[k]} \tag{2.17a}$$

$$G_u\left[k\right] = \left.\frac{\partial g(\vec{s},\vec{u})}{\partial \vec{u}}\right|_{\vec{s}=s[k],\vec{u}=u[k]} \tag{2.17b}$$

Em [17] são apresentadas as equações do algoritmo do FKEC, essas operações estão apresentadas a seguir nas equações 2.18 e 2.19.

ETAPA DE PREDIÇÃO

$$Q\left[k\right] = F_u\left[k\right]\eta F_u\left[k\right] + Q_0 \tag{2.18a}$$

$$\vec{x}\left[k\right] = f(\vec{x}[k-1], \vec{u}\left[k\right]) \tag{2.18b}$$

$$P\left[k\right] = F_x\left[k\right]P[k-1]F_x^T\left[k\right] + Q\left[k\right] \tag{2.18c}$$

ETAPA DE CORREÇÃO

$$\vec{y}\left[k\right] = g(\vec{s}\left[k\right], \vec{u}\left[k\right]) \tag{2.19a}$$

$$R\left[k\right] = G_s\left[k\right]\eta_s G_s^T\left[k\right] + R_0 \tag{2.19b}$$

$$S\left[k\right] = F_u\left[k\right]\eta_u G_u\left[k\right] \tag{2.19c}$$

$$K\left[k\right] = \left(P\left[k\right]H_x^T\left[k\right] + S\left[k\right]\right)\left(H_x\left[k\right]P\left[k\right]H_x^T\left[k\right] + H_x\left[k\right]S\left[k\right] + S^T\left[k\right]H_x^T\left[k\right] + R\left[k\right]\right)^{-1} \tag{2.19d}$$

$$\vec{x}\left[k\right] = \vec{x}\left[k\right] + K\left[k\right](\vec{y}\left[k\right] - H_x\left[k\right]\vec{x}\left[k\right]) \tag{2.19e}$$

$$P\left[k\right] = (\mathrm{I}_{n\times n} - K\left[k\right]H_x\left[k\right])P\left[k\right](\mathrm{I}_{n\times n} - K\left[k\right]H_x[K])^T + K\left[k\right]R\left[k\right]K^T\left[k\right] \tag{2.19f}$$

em que K é o ganho de Kalman, η_u é a matriz de covariância do vetor de controle, η_s é a matriz de covariância do vetor de medidas do sensor $\vec{s}$ e n é o comprimento do vetor de estados.

2.7 CONTROLE PID

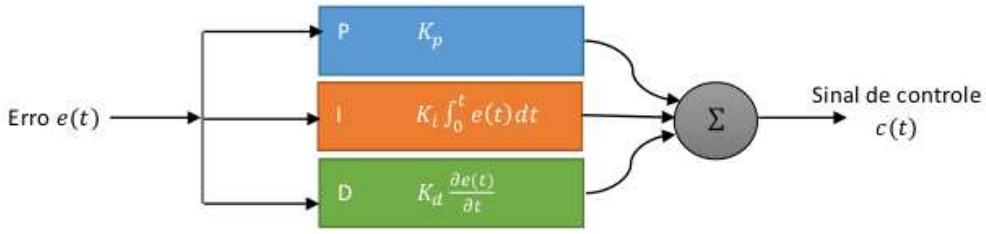


Figura 2.4: Diagrama de blocos do controlador PID contínuo

Um sistema de controle em malha fechada é um sistema em que é feita uma realimentação da resposta do processo sobre controle, denominado planta de controle, para comparação com a entrada de referência com efeito de obter um sinal de controle que se oponha ao crescimento do erro sistema, dado pela diferença entre a entrada de referência e a resposta da planta de controle. As vantagens que podem ser obtidas da realimentação são: estabilização, ou redução dos efeitos de instabilidade, redução da distorção não linear e redução da sensibilidade a ruído e a perturbações [23]. O cálculo do sinal de controle em função do erro é feito pelo controlador.

O controlador PID (proporcional, integrador, derivativo) apresenta na saída um sinal de controle composto pela soma da saída de três controladores: controlador proporcional , controlador integral e derivativo . O controlador proporcional apresenta uma saída proporcional ao erro, controla principalmente o tempo de resposta, em caso em que existe erro de estado estacionário o o ajuste do ganho proporcional não é capaz de anular este erro. O controlador integral apresenta uma saída proporcional a integral temporal do erro, tem como principal efeito a correção do erro do estado estacionário. E o derivativo não contribui diretamente para que a saída siga a entrada, portanto é não de fato um controlador e deve ser utilizado junto com um controlador proporcional ou integral, tem como efeito a oposição a variações na rápidas na resposta do sistema, aumentando o amortecimento do sistema o que diminui o sobressinal e amplitude das oscilações, consequentemente, diminui o tempo de assentamento. A equação 2.20 explicita a equação do sinal de controle c fornecido pelo PID, em que e é o sinal de erro, K_p é o ganho proporcional, K_i é o ganho integrador e K_i é o ganho derivativo [24]. E a figura 2.4 apresenta um diagrama de blocos para o controlador PID contínuo.

$$c(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{\partial e(t)}{\partial t} \tag{2.20}$$

Em sistemas de tempo discreto a equação de controle do PID é obtida pela discretização da integral e da derivada na equação 2.20. A equação 2.21 apresenta a equação do controlador PID discreto [24], em que T_s é o período de amostragem do erro do sistema.

$$c[k] = c[0] + K_p e[k] + K_i T_s \sum_{0}^{k-1} e[k] + K_d \frac{e[k] - e[k-1]}{T_s} \tag{2.21}$$

O controlador PID apresenta a vantagem de possuir apenas três parâmetros para ajuste, o que viabiliza

a estimação desses parâmetros por métodos experimentais quando não há possibilidade de uma abordagem analítica devido à ausência de um modelo matemático da planta de controle [24]. Os principais critérios de desempenho no ajuste de controladores PID são:

- o erro de estacionário (ess) definido como a diferença entre a entrada de referência (Ref) e a saída em regime permanente;
- o tempo de assentamento definido como o tempo que a resposta demora estar confinada em uma faixa de tolerância em torno do valor de estado estacionário;
- e o sobressinal definido como a diferença entre o valor de pico da resposta do sistema a resposta de estado estacionário.

A figura 2.5 ilustras os parâmetros de desempenho.

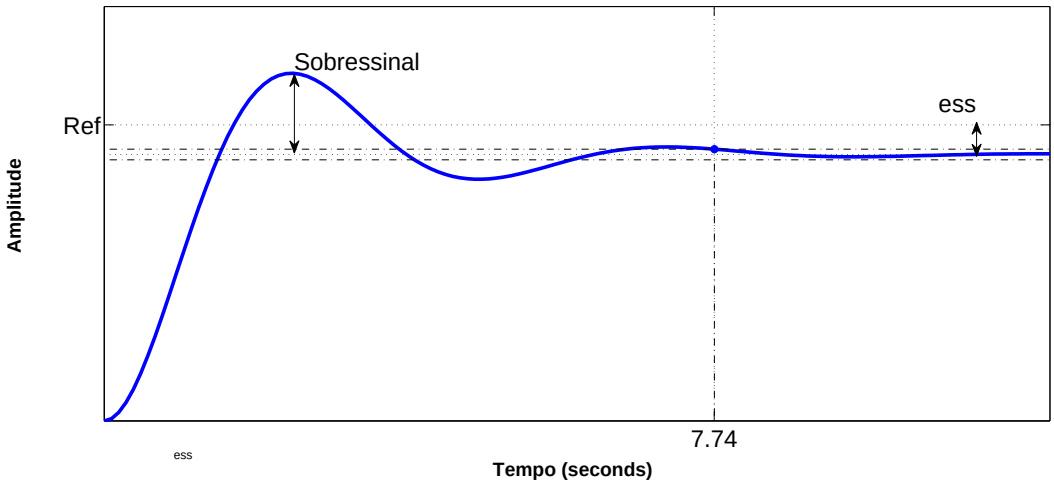


Figura 2.5: Desempenho de controladores PID

3 DESENVOLVIMENTO

"It is not in doing what you like, but in liking what you do that is the secret of happiness."

- J.M. Barrie, Peter Pan

3.1 INTRODUÇÃO

O aparato de teste desenvolvido neste trabalho possui uma estrutura projetada para fixação de um helimodelo que proibi todos os movimentos de translação e permitir todos os movimentos de rotação. Um sistema de detecção de atitude composto fisicamente por sensores embarcados, sistema de interface e computador, fornece estimativas da atitude do helicóptero em tempo real. O computador além de executar o algoritmo de estimação da atitude é responsável por receber e executar o algoritmo de controle a ser testado. O sistema de interface é encarregado de ler os sensores, enviar os dados lidos para o computador, receber os sinais de controle do computador e enviar os sinais de controle para os motores, garantindo o período de amostragem estipulado para leitura dos sensores e atualização dos sinais de controle.

O helimodelo utilizado no projeto é um modelo elétrico comercial, modelo *BETL CPX* do fabricante *E-SKY*®, mostrado na figura 3.1. Trata-se de um modelo leve (0.670Kg), pequeno (650mm), de fácil montagem. Possui um motor de corrente contínua (modelo EK5-0005) acionado por um controlador eletrônico de velocidade (modelo EK1-0350) que aciona o rotor principal e o rotor de cauda, e quatro servo-motores (modelo EK2-0508), um servo para inclinar as pás do rotor de cauda, um servo para inclinar as pás do rotor principal, e dois servos para alterar a orientação do plano de rotação do rotor principal. O *BETL CPX* possui um sistema de controle do movimento de guinada, composto por um giroscópio (EK2-0704B) posicionado de forma a medir a velocidade angular do rotor de cauda, e um controlador proporcional que atua nos ângulos do rotor de cauda para compensar os movimentos não acionados pelo comando pedal.

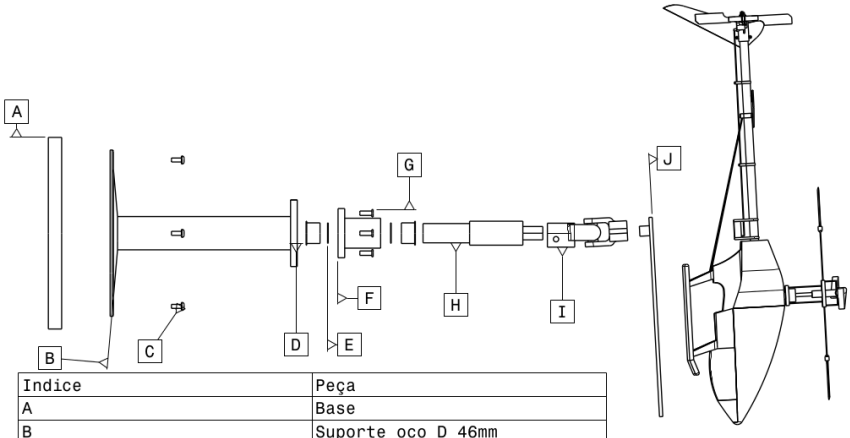
Uma característica importante deste helimodelo é que ele possui todos os mecanismos de controle de angulação das pás do rotores existentes em um helicóptero convencional (comandos: coletivo, cíclico e pedal), o que aproxima sua dinâmica do helicóptero convencional. Além disso, este helimodelo utiliza um motor elétrico que não requer nenhum tipo de gás combustível e, portanto, não produz quaisquer gases de escape durante o seu funcionamento, o que o torna ideal para realizações de testes em ambientes fechados.

Figura 3.1: Helicóptero elétrico *Belt CPX* fabricado pela *E-SKY*®

3.2 PROJETO DA ESTRUTURA MECÂNICA DA PLATAFORMA

A estrutura mecânica da plataforma desenvolvida é divida em três partes: suporte, articulação e palco. A figura 3.2 apresenta uma foto da primeira versão da estrutura mecânica da plataforma. A figura 3.3 apresenta um desenho de conjunto em vista explodida da estrutura mecânica da plataforma.

Figura 3.2: Foto da estrutura mecânica da plataforma



Indice	Peça
A	Base
B	Suporte oco D 46mm
C	Parafuso M8
D	Bucha 15mm
E	Anél metálico
F	Rolamento LFM 30
G	Parafuso D 6.6 m
H	Eixo de conexão (rolamento-junta)
I	Junta universal
J	Apoio do helicóptero

Figura 3.3: Desenho de conjunto da estrutura mecânica da plataforma

O suporte é a base inferior da plataforma, foi projetado para viabilizar a fixação da plataforma no local de trabalho, elevar o helicóptero para que não entre em contato com a base durante os movimentos e possibilitar a passagem de cabos no seu interior para evitar que se enrolem na plataforma. O suporte possui uma base inferior oval com quatro furos para parafusos, um eixo cilíndrico vertical oco no centro para permitir a passagem de fios e aumentar a altura da plataforma, e uma base superior quadrada para fixação da articulação da plataforma.

A articulação é a parte móvel da plataforma, montada sobre o suporte, responsável por permitir que helicóptero realize os movimentos de rotação. Para permite o movimento de guinada a articulação possui um rolamento de eixo linear com diâmetro interno de 30mm instalado por um mancal na base superior do suporte. O rolamento é conectado por um eixo de transferência a uma junta com dois graus de liberdade que permite os movimentos de arfagem e rolagem. Optou-se por um eixo de transferência oco para permitir a passagem de cabos em seu interior, o eixo foi fabricado em alumínio trefilado.

Na primeira versão da plataforma optou-se por utilizar uma junta Cardan automotiva de aço, mas o seu material foi responsável por pertubação nas medidas do magnetômetro utilizado para medir o campo magnético terrestre. Como alternativa optou-se por projetar uma junta de plástico ABS que possa ser fabricada em impressora 3D. A nova junta foi projetada para, além de apresentar dois graus de liberdade, permitir instalação de potenciômetros em seus eixos de rotação para que funcionem como sensores de rotação, e permitir que a placa *SEN-10724* da *Sparkfun* seja instalada no seu centro de rotação para reduzir os efeitos de translação nas medidas do campo gravitacional feitas pelo acelerômetro. A nova junta, apresentada na figura 3.4, é formada por um garfo inferior que encaixa no eixo de transferência, um garfo superior que encaixa no palco, uma cruz central com espaço para a placa de sensores. A cruz é encaixada nos dois garfos por meio de rolamentos de eixo de 8mm modelo *ABEC7* para que exista liberdade de movimento de rotação com atrito reduzido.

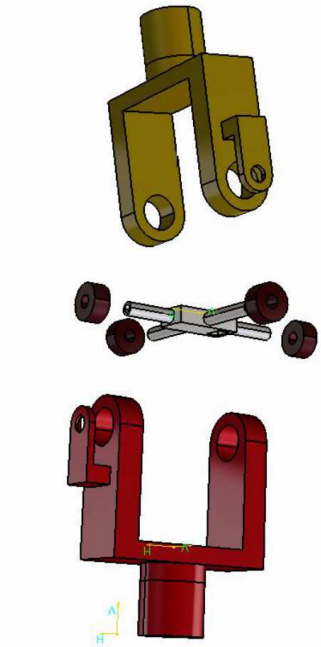


Figura 3.4: Desenho de conjunto da junta projetada

O palco é a parte da plataforma onde o helicóptero é diretamente fixado, consiste de uma placa de acrílico fixada no garfo superior da juntas.

3.3 SENSORES: PARÂMETROS DE CONFIGURAÇÃO E CALIBRAÇÃO

A placa de sensoriamento usada na plataforma é a *SEN-10724* fabricada pela *Sparkfun*, apresentada na figura 3.5 que integra um acelerômetro triaxial ADX345, um giroscópio triaxial ITG-3200 e um magnetômetro triaxial HMC5843. O giroscópio mede a velocidade angular do helimodelo, essa medida é usada na propagação temporal da atitude estimada. O acelerômetro mede o oposto do vetor campo gravitacional terrestre e o magnetômetro mede o vetor campo magnético terrestres, esses vetores são usados na estimação da atitude com algoritmo TRIAD. A estimação pelo método TRIAD e a propagação temporal são combinadas por um filtro de Kalman para obter-se uma estimativa mais precisa. A *SEN-10724* é instalada para mover-se junto com o helicóptero, fazendo com que os eixos de medida mantenham-se constante em relação ao helimodelo em qualquer movimento. As medidas são feitas segundo um sistema de referência *b-frame* com origem no centro de rotação da plataforma.

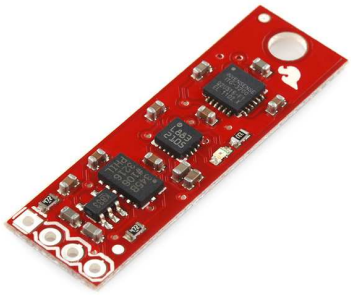


Figura 3.5: Placa de sensores *SEN-10724* fabricada pela *Sparkfun*

O ADLX345 é um acelerômetro com resolução de 13 bits e faixa de medida de $\pm 6g$. O seu sinal de saída é um vetor de 3 elementos de 16 bits em complemento de dois, em que cada elemento representa a medida em um eixo de medida. O ADLX345 possui um modo de operação de baixo consumo e um modo de operação ativo para medidas. Para iniciar o modo ativo deve-se escrever o valor binário 1 no terceiro bit menos significativo do registrador de configuração cujo o endereço é Ox2D.

A calibração do acelerômetro corrigi o vetor de viés $\vec{b} = [b_x \ b_y \ e_b,]$ e a diferença entre o fator de escala de cada eixo. A equação 3.1 apresenta a relação da saída do acelerômetro não calibrada $\vec{a}_e$ com a medida livre de erro $\vec{a}$.

$$\vec{a}_e = \begin{bmatrix} a_{e,x} \\ a_{e,y} \\ a_{e,z} \end{bmatrix} = \begin{bmatrix} k_x & 0 & 0 \\ 0 & k_y & 0 \\ 0 & 0 & k_z \end{bmatrix} \vec{a} + \vec{b} \tag{3.1}$$

Se o fator de escala de todos os eixos de medida fossem iguais, a diferença entre o valor máximo e

o viés seria igual para todos os eixos. Para que os três eixos apresentem o mesmo fator de escala, cada eixo deve ser multiplicado pelo inverso da diferença entre o valor máximo medido e o viés, e pelo fator de escala de desejado. Como as medidas do acelerômetros são normalizadas durante o algoritmo TRIAD, o fator de escala aplicado às medidas é irrelevante, por esse motivo, o fator de utilizado foi 1. A equação 3.2 apresenta a equação para obtenção de um vetor de medidas calibradas $\vec{a}_c$ a partir do vetor de saída do acelerômetros $\vec{a}_e$.

$$\vec{a}_c = \begin{bmatrix} \frac{1}{\max(\vec{A}_x)-b_x}(a_{e,x} - b_x) \\ \frac{1}{\max(\vec{A}_y)-b_y}(a_{e,y} - b_y) \\ \frac{1}{\max(\vec{A}_z)-b_z}(a_{e,z} - b_z) \end{bmatrix} \tag{3.2}$$

O ITG-3200 é um giroscópio triaxial dedicado para jogos,*3D mouses* e aplicações com controle remotos 3D. O sinal de saída é composto por três elementos de 16 bits, em que cada elemento representa a velocidade angular em torno de um dos eixos de medida. O fator de escala, a frequência de amostragem e o filtro passa baixa aplicado à saída devem ser configurados pelo registrador DLPF (endereço 0x22). As configurações utilizadas neste trabalho são: fator de escala $(1/14.375)^\circ/s$, $1kHz$ de taxa de amostragem e frequência de corte de $42Hz$.

A calibração do giroscópio subtrai o vetor de viés $\vec{b}_g$ e aplica o fator de escala nominal $(1/14.375)^\circ/s$ às medidas sem viés.

$$\vec{g}_c = \frac{1}{14.375}\left(\vec{g}_e - \vec{b}_g\right) \tag{3.3}$$

O HMC5843 é um magnetômetro com sensibilidade e linearidade projetadas para permitir medidas do módulo e direção do campo magnético terrestre. Sua saída é formada por um vetor de 3 elementos de 16 bits, em que cada elemento corresponde à componente do campo magnético medida em um eixo. O modo de medida continua é configurado escrevendo o valor binário 0 nos dois bits menos significativos do registrador Mode Register (endereço 0x02).

A calibração do magnetômetro quando deseja-se medir o campo magnético da terra deve considerar além do viés e do erro de fator de escala inerentes do magnetômetro a distorção causada pelo ambiente. A distorção é corrigida junto com o erro de fator de escala por uma matriz de transformação linear. A Equação 3.4 apresenta a equação de calibração do magnetômetro.

$$\vec{m}_c = M_s(\vec{m}_e - \vec{b}_m) \tag{3.4}$$

em que M_s é a matriz de correção da distorção, $\vec{m}_e$ é a medida não calibrada, $\vec{b}_m$ é o viés e $\vec{m}_c$ é a medida calibrada.

3.3.1 Estimação dos parâmetros de calibração

Essa seção apresenta o procedimento experimental que deve ser realizado para estimar os parâmetros de correção dos erros na saída dos sensores, conforme discutido na seção 3.3. Esses parâmetros variam

cada vez que a placa *SEN-10724* é iniciada, por esse motivo antes de cada experimento com o sistema de detecção de atitude desenvolvido é necessário realizar a estimação dos parâmetros de correção.

A seção 4.3 apresenta os resultados obtidos para uma realização dos procedimentos de calibração.

ACELERÔMETRO

O procedimento experimental realizado para estimação do viés do acelerômetro foi extraído de [25]. Em que primeiro posiciona-se a placa *SEN-10724* com o eixo de medida x na vertical apontando para baixo, nesse estado a direção do eixo x deve coincidir com a direção do campo gravitacional, o que deve ser confirmado observando que apenas o elemento x do vetor de medida não é próximo de zero. Em seguida gira-se a placa em 360° em torno do eixo y com cuidado para não provocar movimentos de translação. O mesmo procedimento deve ser repetido iniciando-se com o eixo y na vertical apontando para baixo e realizando rotações em torno dos eixos x e z.

Durante o experimento cada eixo de medida deve passar uma vez pelas orientações: vertical apontando para cima e vertical apontado para baixo, condições em que a medida no eixo corresponde a aceleração gravitacional e ao oposto da aceleração, respectivamente. Assim, o valor máximo medido em cada eixo corresponde ao módulo da força específica de atração gravitacional e o valor mínimo medido corresponde ao oposto do módulo da força específica de atração gravitacional, ambos somados de um mesmo erro constante de viés. Na ausência do viés a média entre o valor máximo e mínimo medido em qualquer um dos eixos seria zero. Com efeito do viés a média entre o valor máximo e e o valor mínimo medido é igual ao viés. O viés de cada eixo é obtido da média entre o valor máximo e mínimo das medidas no eixo durante o experimento, a equação 3.5a enfatiza a operação matemática realizada.

$$\vec{b}_a = \frac{\max(A) + \min(A)}{2} \tag{3.5a}$$

$$A = \begin{bmatrix} \vec{A}_x \\ \vec{A}_y \\ \vec{A}_z \end{bmatrix} = \begin{bmatrix} \vec{a}_e[0] & \vec{a}_e[1] & \vec{a}_e[2] & \dots \end{bmatrix} \tag{3.5b}$$

Como discutido na seção 3.3, se o fator de escala de todos os eixos de medida fossem iguais, após a subtração do viés, o valor máximo medido serial igual para os três eixo. Para corrigir o erro de fator de escala, as medidas devem ser multiplicadas por um fator de correção de escala obtido para cada eixo pela equação 3.6, em que I_{max} é o valor máximo medido no eixo durante no experimento.

$$K_i = \frac{1}{I_{max} - b_a} \tag{3.6}$$

GIROSCÓPIO

O procedimento experimental realizado para estimar o vetor de viés do giroscópio consiste em deixar o giroscópio imóvel por um longo período, por exemplo 10 min, e registrar todas as medidas feitas neste período. Com o giroscópio parado, pode-se considerar que a saída observada é igual a soma do viés com um ruído aleatório de média zero. O viés foi calculado pela média das amostras obtidas.

MAGNETÔMETRO

O viés e a matriz de correção das medidas do magnetômetro podem ser obtidos utilizando a ferramenta com interface gráfica fornecida em [25].

3.4 INTERFACE DE AQUISIÇÃO DE DADOS E ATUAÇÃO

A plataforma de teste desenvolvida utiliza um computador para executar o algoritmo de estimação da atitude a partir das medidas do acelerômetro, giroscópio e magnetômetro, executar o algoritmo de controle em teste e gerar os sinais de controle do acionamento dos motores do helimodelo. Um sistema de interface é necessário para realizar a leitura dos sensores e enviar os dados lidos para o computador, receber do computador os comandos de acionamento dos motores e acionar os motores de acordo com os comandos recebidos. Para o fim de testar controladores digitais é necessário que o período de amostragem dos sinais medidos pelos sensores e o período de atualização dos sinais enviados para os atuadores sejam sincronizados e constante, por essa razão o sistema de interface deve operar em tempo real para que seja capaz de realizar a sincronização e controlar o período de amostragem. A taxa de amostragem adotada foi 50Hz que corresponde à frequência do sinal PWM de acionamento dos servo-motores do helimodelo, em outras palavras, corresponde à taxa máxima de atualização da posição dos servo-motores.

A tabela 3.1 apresenta o pacote de dados enviado da interface para o computador composto pelas medidas de 16 bits, em que A, G e M simbolizam medidas do acelerômetro, giroscópio e magnetômetro, respectivamente, e os índices x, y e z referem-se aos eixos de medida. A tabela 3.2 apresenta o pacote de dados enviado do computador para a interface composto pelos sinais de acionamento dos motores, os sinais de acionamento são representados por variáveis do tipo float com dois dígitos de parte inteira e dois dígitos de parte fracionária.

Tabela 3.1: Pacote de dados enviado da interface para o computador

Dados	M_z	M_y	M_x	G_z	G_y	G_x	A_z	A_y	A_x
Bytes	17-16	15-14	13-12	11-10	9-8	7-6	5-4	3-2	1-0

Tabela 3.2: Pacote de dados enviados do computador para a interface

Dados	Rotor de cauda	Servo 1	Servo 2	Servo 3	Servo 4
Bytes	19-16	15-12	11-8	7-4	3-0

Neste trabalho foi desenvolvida uma versão de interface com o *Arduino* e outra versão com *Raspberry Pi*. A versão em *Arduino* permite um controle mais preciso dos instantes de início de cada tarefa, pois a alternância entre funções não está sujeita a decisão de escalonadores de sistemas operacionais. Mesmo que não sejam necessários recursos de sistemas operacionais, a versão em Raspberry Pi utiliza o sistema operacional *Raspbian* com *framework Xenomai*, porque embora a alternância entre tarefas esteja sujeita a escalonador de processos, a frequência de processamento mantém os atrasos toleráveis, e a API do *Xenomai* facilita o desenvolvimento, além de disponibilizar um sistema operacional em terminal que pode ser útil para trabalhos futuros por exemplo: integração do algoritmo de estimativa da atitude e de controle

na *Raspberry Pi* e criação de uma interface amigável com o usuário.

A proposta inicial foi a versão em Arduino, mas a interpretação errada do resultado de testes levou a conclusão de que o *Arduino* não seria capaz de executar as tarefas da interface com desempenho satisfatório. Por esse motivou-se iniciou-se o desenvolvimento *Raspberr Pi*. Após a conclusão da versão com *Raspberry Pi* identificou-se o corrigiu-se o erro cometido na versão em *Arduino*. Como as duas versões atendem os requisitos do projeto e apresentam características diferentes, optou-se por apresentar as duas versões neste relatório.

3.4.1 Interface baseada em microcontrolador

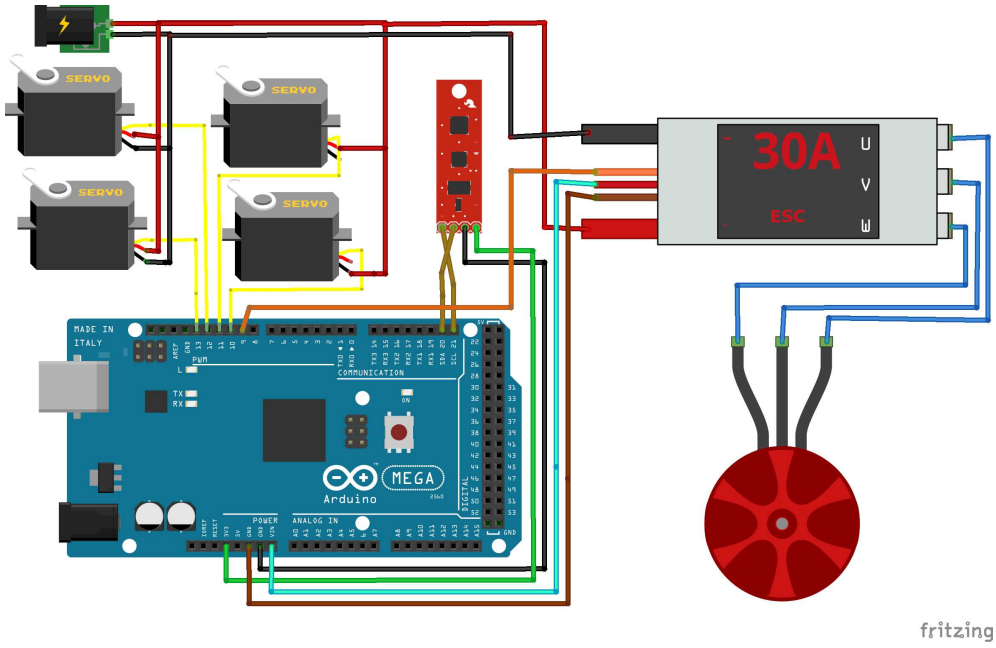


Figura 3.6: Esquema de montagem da interface baseada em *Arduino*

A primeira versão da plataforma foi desenvolvida em *Arduino Mega 2560* visando uma prototipagem rápida para definição dos meios de comunicação e formato dos pacotes de dados enviados e para antecipar o começo do desenvolvimento do algoritmo de estimação de atitude que depende das medidas dos sensores. O Anexo I apresenta o código de programação desenvolvido.

O *Arduino Mega 2560* mostrou-se interessante para o projeto por possuir I/O dedicada para barramento I^2C, suporte para comunicação serial com computador via USB e uma quantidade de canais PWM suficiente para controlar os cinco motores do helimodelo *BELT CPX*. A figura 3.6 apresenta um diagrama de conexão do *Arduino* com a placa de sensores e os motores. A placa de sensores *SEN-10724* é conectada ao *Arduino* pelos pinos de barramento I2C e alimentada pelos pinos +3V3 e GND. Os sinais de acionamento são enviados para os motores pelos pinos dedicados para PWM, pois os servo-motores são controlados por sinais PWM e o motor DC é acionado por controlador eletrônico de velocidade que recebe como entrada de referência um sinal PWM. Os motores são alimentados pelo sistema de alimentação interno do *BELT CPX*.

Algoritmo 1 - Processos da interface implementada em Arduino	
Aquisição	**Atuação**
Aguarda *Idle Mode* ser interrompido pelo *Timer4*	Aguarda *Idle Mode* ser interrompido pelo *Timer4*
Leitura do acelerômetro	Recebe os sinais de comando enviados pelo computador
Leitura do giroscópio	Atualiza o acionamento do rotor principal
Leitura do magnetômetro	Atualiza o acionamento do servo de inclinação das pás do rotor principal
Envio das leituras para o computador	Atualiza o acionamento do servo de alteração do ângulo de batimento longitudinal
Coloca o *Arduino* em *Idle Mode*	Atualiza o acionamento do servo de alteração do ângulo de batimento lateral
	Atualiza o acionamento do servo de inclinação das pás do rotor de cauda.
	Coloca o *Arduino* em *Idle Mode*

O *Arduino* não possui suporte para multi-processamento, não sendo possível executar simultaneamente todas as tarefas da interface de aquisição de dados e atuação. A solução adotada para executar todas as tarefas da interface foi dividi-las em dois processos, aquisição e atuação, de forma que um processo executa uma iteração de suas tarefas e em seguida o outro processo executa uma iteração de suas tarefas. Uma iteração do processo aquisição é responsável por ler os dados dos sensores e enviá-los para o computador. Uma iteração do processo atuação é responsável por ler os sinais de comandos enviados pelo computador e acionar os motores com os comandos recebidos. Para garantir o período de amostragem constante, o intervalo entre duas iterações de um mesmo processo é controlado para que seja igual ao período de amostragem T_s. Se uma iteração do processo aquisição começa em t_a, a iteração do processo atuação começará em $t_b = t_a + t_s/2$, e a próxima iteração de aquisição começará em $t_a + T_s$ e a próxima iteração do processo atuação começará em $t_b + T_s$. Para que a plataforma opere com período de amostragem T_s é necessário que nenhum dos processos demore mais do que metade do período de amostragem para executar uma iteração. O algoritmo 1 apresenta a estrutura dos processos de aquisição e atuação.

Entre a leitura dos sensores e o próximo acionamento dos motores existe um atraso de meio período de amostragem. Devido a este atraso é possível que a leitura dos sensores em uma iteração do processo de aquisição interfira na atuação dos motores na iteração seguinte do processo de atuação, se o tempo total entre a leitura dos sensores e a disponibilização do sinal de controle pelo computador for inferior ao atraso de meio período de amostragem. Em outras palavras, é necessário que o computador disponibilize o comando de controle antes do começo da iteração de atuação.

A programação do *Arduino* possui obrigatoriamente duas funções: *setup* e *loop*. A função *setup* é a primeira a ser executada, onde a comunicação i2c é iniciada, a comunicação serial é configurada para *baud-rate* de 115200B/s, a placa *SEN-10724* é configurada e o registrador de tempo utilizado para sincronização é configurada. A função *loop* é um laço de repetição infinita onde as funções de operação do *Arduino* são programadas.

O recurso utilizado para a alternância sincronizada entre os processos é a interrupção por registrador de tempo. A iteração de cada processo encerra com o comando para que o *Arduino* entre no estado *idle mode*, estado em que a execução é suspensa, mas os dados recebidos pela comunicação serial não serão perdidos e o acionamento dos motores não é interrompido. O *Arduino* é configurado para sair do *idle mode* quando houver uma interrupção do *Timer4*. Ao retornar a execução uma iteração do processo diferente do anterior começa a ser executada. O *Timer4* é configurado para gerar interrupções periódicas com período $T_s/2$, para que a cada $T_s/2$ a alternância entre os processos acorra a cada $T_S/2$ e o intervalo entre as repetições dos processos seja T_s.

O *Timer4* possui um contador de 16 bits controlado pelo clock interno de 16Mhz com divisores de clock disponíveis. As configurações do *Timer4* são armazenadas nos registradores `TCCR4A` e `TCCR4B`. Configurou-se o *Timer4* para gerar interrupções por comparação, utilizar divisão de clock com razão 1:64 e reiniciar contagem após a interrupção.

A interrupção por comparação é gerada quando o valor no registrador de contagem `TCTN4` iguala-se ao registrador de comparação `OCR4A`. Para gerar interrupções periódicas com intervalos de $T_s/2$, o contador `TCTN4` deve ter valor inicial em zero, e o registrador de comparação deve ser igual a:

$$[\mathrm{OCR4A}] = \frac{(T_s/2)\cdot f}{k} - 1 \tag{3.7}$$

em que f é a frequência de clock e k é a fator de divisão de clock. Por exemplo, para obter um interrupções a cada 10ms utilizando o divisor de clock com fator 64, o registrador de comparação deve ser igual a 2499.

A análise de desempenho da interface de aquisição e atuação implementada em *Arduino* é apresentada na seção 4.2.2.

3.4.2 Interface baseada em microcomputador

A segunda versão da interface de aquisição de dados e atuação foi implementada com a *Raspberry Pi* por apresentar compatibilidade com sistema operacional em tempo real com suporte para multiprocessamento, o que permite que as tarefas da interface sejam execução de forma pseudo-paralela das tarefas da interface. A *Raspberry Pi* utilizada é do modelo B, com sistema operacional *Raspbian* e *framework Xenomai* para dar suporte à aplicações em tempo real. Para comunicação entre a Raspberry Pi e o computador é utilizado um conversor usb-serial *FTRS232 Breakout* fabricado pela *Sparkfun*. A figura 3.7 apresenta um diagrama de conexão da *Raspberry Pi* com a placa de sensores *SEN-10724*, com o conversor usb-serial e os motores.

A solução adotada para implementação das funções de operação da interface de aquisição e atuação foi dividias em três tarefas: leitura, recebimento e atuação. A tarefa leitura é composta pelas funções de leitura dos sensores e envio dos dados lidos para o computador. A tarefa de recebimento é composta pelas funções de aguardar e registrar os sinais de comando enviados pelo computador. E a tarefa de atuação é composta pelas funções de enviar os comandos recebidos do computador para os motores. O algoritmo 2 apresenta os procedimentos das tarefas.

A aplicação, desenvolvida em *Linguagem C*, foi implementada em um único processo de múltiplas

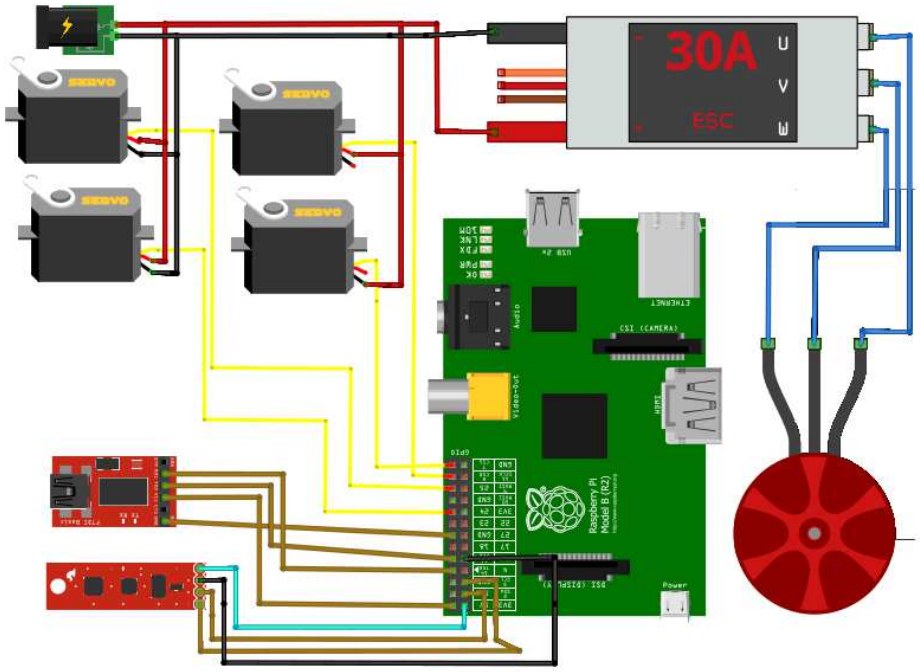


Figura 3.7: Esquema de montagem da interface baseada em *Raspbery Pi*

tasks. *Tasks* são sub-unidades de processamento menores do que um processo, que compartilham espaço de memória entre si e podem ser sincronizadas pelos recursos do sistemas disponibilizados pelo Xenomai. Foi criada um *task* para cada uma das três tarefas. O Anexo II apresenta o código de programação da aplicação desenvolvida.

A comunicação com a placa de sensores pelo barramento I2C foi implementada utilizando a biblioteca *wiringPiI2C.h* cuja a instalação e funções disponíveis estão descritas em [26].

A comunicação serial com o computador foi implementada utilizando a biblioteca *wiringSerial.h* descrita em [27].

A *Raspberry Pi* não possuí pinos de saída PWM controlados por hardware dedicado disponíveis para os cinco motores do helimodelo. Uma alternativa é implementar um sinal PWM por software utilizando os recursos de temporização disponíveis pelo *Xenomai* para controlar a saída de um pino para que se comporte como um sinal de onda quadrada de 50Hz com *duty cycle* desejado. A biblioteca *servo.h*, disponível em [28], disponibiliza funções de acionamento eficaz de servo-motores por sinal PWM.

Para a sincronização das *tasks* foram criados dois alarmes periódicos, *Alarme_1* e *Alarme_2*. Alarmes são mecanismos de sincronização que geram sinais agendados que acordam as *tasks* em espera destes sinais, as funções de utilização de alarmes estão disponíveis na biblioteca *alarm.h* descrita em <http://www.xenomai.org/documentation/trunk/html/api/alarm_8h.html>. Uma *task* que espera um alarme é retirada do estado de execução para o estado de espera e retorna para execução assim que o sinal do alarme em espera ocorrer. O Alarme *Alarme_1* é usado para sincronizar as *tasks* de aquisição e atuação e o sinal de *Alarme_2* é usado para controlar a *task* de recebimento, os dois alarmes possuem período de repetição

Algoritmo 2 - Descrição das tarefas da interface implementada em Raspberry-Pi
Leitura
Aguarda *ALARME_1*
Lê acelerômetro
Lê giroscópio
Lê magnetômetro
Envia dos dados lidos para o computador
Recebimento
Aguarda *ALARME_2* Recebe os dados enviados pelo computador
Converte dados recebidos de *string* para *float*
Atuação
Aguarda *ALARME_1*
Atualiza o acionamento do rotor principal
Atualiza o acionamento do servo que altera a inclinação das pás do rotor principal
Atualiza o acionamento do servo que altera o ângulo de batimento longitudinal
Atualiza o acionamento do servo que altera o ângulo de batimento lateral
Atualiza o acionamento do servo que altera a inclinação das pás do rotor de cauda

igual à taxa de amostragem, assim a leitura dos sensores e a atualização do comando dos motores são realizadas com a taxa de amostragem desejada e de forma aproximadamente simultânea, pois solicitam ao escalonador a mesma janela de tempo de execução, entretanto a execução está sujeita aos atraso de atendimento do escalonador que não é controlado pela aplicação. As três *tasks* são formadas por um laço de repetição infinita que começa com um comando de espera por sinal de alarme. De forma que a cada sinal de alarme uma nova iteração do laço é executada e o intervalo entre as iterações seja igual ao período de amostragem.

O *Alarme_2* possui um atraso em relação ao *Alarme_1* para que exista um tempo entre a leitura do sensor e a leitura do comando enviado pelo computador, este tempo é necessário para que o computador execute o algoritmo de estimação da atitude e o algoritmo de controle envie os sinais de comando. O atraso deve ser menor do que período de amostragem para que o sinal de controle calculado no instante k seja enviado para os atuadores no instante $k + 1$. Optou-se por utilizar um atraso de 10ms igual à metade do período de amostragem, sobrando metade do período de amostragem para que os dados recebidos do computador sejam lidos e armazenados antes da próxima atualização do comando de acionamento dos motores.

A função *main* inicia a comunicação I2C e a comunicação serial com *baud-rate* de 115200B/s. Em seguida configura a placa de sensores *SEN-10724* e a configuração dos servo-motores. A placa de sensores é configurada conforme apresentado na Seção 3.3. Os sinais PWM enviados para os servo-motores são configurados para servo-motores controlados por PWM com *duty-cycle* que varia de $1ms$ à $2ms$ e posição que varia de 0° a 90°. Em seguida, na função *main*, são criados os alarmes. E por ultimo são criadas as as três *tasks*.

A análise de desempenho da interface de aquisição e atuação implementada em *Raspberry Pi* é apre-

sentada na seção 4.2.3.

3.5 SISTEMA DE DETECÇÃO DA ATITUDE

O sistema de detecção da atitude implementado na plataforma é responsável por fornecer a uma taxa de amostragem de 50Hz estimativas da orientação do helimodelo representada em ângulos de Euler. Esta seção descreve as operações matemáticas realizadas com os dados lidos dos sensores até a obtenção dos ângulos de Euler. Os algoritmos foram implementados em *Matlab*. O Anexo III apresenta o *script FKEC.m* desenvolvido.

O Sistema de Navegação inercial utiliza medidas de velocidade angulares para fazer a propagação temporal da atitude de acordo com os modelos cinemáticos, e medidas do campo gravitacional e do campo magnético terrestre para estimar a atitude com o algoritmo TRIAD. Um Filtro de Kalman Estendido Correlato, formulação do filtro de Kalman usado para modelos não lineares que considera a correlação entre ruídos, faz a fusão entre a estimativa obtida pelas equações cinemáticas e a estimativa obtida pelo método TRIAD. As estimativas são expressas no sistema de referência *n-frame* com orientação fixa em relação a parte imóvel da plataforma, enquanto os sensores são instalados para fornecer vetores representados no sistema de coordenadas *b-frame* fixo em relação ao helimodelo, de forma que a em relação ao *n-frame* a orientação do helimodelo é igual à orientação do *b-frame*.

Neas sessões seguintes considera-se que os sensores medidos já estão calibrados. As operações de calibração descritas na Seção 3.3 é a primeira operação a ser realizada com os dados obtidos.

3.5.1 Método TRIAD

A orientação do helicóptero pode ser obtida calculando-se a orientação do *b-frame* em relação ao sistema *n-frame*. O método TRIAD é utilizado para calcular a orientação do *b-frame* através da representação dos vetores campo gravitacional e campo magnético representados nos sistemas *b-frame* e *n-frame*. A representação desses vetores *b-frame* mudam com a orientação do helicóptero e precisam ser medidas a cada nova estimativa, por esse motivo o acelerômetro e o magnetômetro são instalados com eixos de medida coincidentes com as bases de *b-frame*. Enquanto a representação desses vetores em *n-frame* precisa ser medida apenas uma vez e a mesma medida pode ser utilizada para todas as estimativas, pois o sistema *n-frame* permanece constante em relação à Terra.

Segundo o método TRIAD descrito na seção 2.5, a matriz de rotação MR_b^n que relaciona os sistemas de coordenas *n-frame* e *b-frame* é estimada pela equação 2.9 aplicada aos vetores aceleração gravitacional normalizada G e campo magnético normalizado M observados nos sistemas *n-frame* e *b-frame*. Os vetores $\vec{i}_n$, $\vec{j}_n$ e $\vec{k}_n$ são calculados apenas uma vez no início do processo de estimação. Os vetores $\vec{i}_b$, $\vec{j}_b$ e $\vec{k}_b$ são calculados a cada iteração com os últimos vetores aceleração gravitacional e campo magnético medidos pelo acelerômetro e magnetômetro, respectivamente.

3.5.2 Filtro de Kalman Estendido Correlato

O Filtro de Kalman implementado apresenta na saída um vetor de estado $\vec{q}_x$ composto pelo quatérnio de atitude. A representação em quatérnios unitários foi escolhida por representar a orientação de forma biunívoca sem singularidades para descrição de movimentos fisicamente realizáveis. Apesar de equivalentes, o vetor de estado é calculado pelo filtro de Kalman e o vetor de comparação é obtido da transformação da matriz de rotação calculada pelo método TRIAD para quatérnios [20]. O vetor de controle $\vec{u}$ é formado pelas medidas de velocidade angular. E o vetor de comparação $\vec{q}_y$ é formado pela estimativa do quatérnio de atitude obtida pelo algoritmo TRIAD e um erro de pseudo-observação para manutenção da norma unitária do quatérnio de atitude [29]. A equação 3.8 enfatiza os vetores estimados. Como o erro de pseudo-observação em $\vec{q}_y$ é calculado diretamente do vetor de estado, existirá correlação entre o ruído da predição do vetor de estado e o ruído de correção que depende do vetor de comparação.

$$\vec{q}_x = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix} \quad \vec{q}_y = \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \\ e_{pseudo} \end{bmatrix} \quad \vec{u} = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} = \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{3.8}$$

O modelo matemático do vetor de estado, em função do vetor de controle e do vetor de estado anterior, corresponde ao modelo cinemático não linear em tempo discreto da atitude em quatérnios, em função da velocidade angular e da amostra anterior da atitude em quatérnios, apresentado na equação 2.8a. A equação 3.9 apresenta a relação entre os vetores de estado, de comparação e de controle.

$$\vec{q}_x[k] = f(\vec{q}_x[k-1], \vec{u}[k]) = \left[\cos(\frac{1}{2}\left|\vec{\vec{u}}[k]\right| \Delta t)\mathrm{I}_{4x4} + \frac{\sin(\frac{1}{2}|\vec{u}[k]|\,\Delta t)}{|\vec{u}[k]|\,\Delta t} W[k]\Delta t\right] \vec{q}_x[k-1] \tag{3.9a}$$

$$W = \Omega(\vec{u}) = \begin{bmatrix} 0 & u_z & -u_y & u_x \\ u_z & 0 & u_x & u_y \\ u_y & -u_x & 0 & u_z \\ -u_x & -u_y & -u_z & 0 \end{bmatrix} \tag{3.9b}$$

$$\vec{q}_y[k] = h(\vec{q}_x[k]) = \begin{bmatrix} \vec{q}_x \\ 1 - |\,|q_x| \end{bmatrix} \mathrm{I}_{4x4}\vec{q}_x[k] \tag{3.9c}$$

As matrizes de atualização F_x e F_u e a matriz de comparação H_x são calculadas a cada iteração do filtro de Kalman. As matrizes F_x e F_u, apresentadas nas equações 3.10 e 3.11, são as jacabianas de $f(\vec{x}, \vec{u})$ em relação a $\vec{x}$ e a $\vec{u}$, respectivamente, avaliadas em $\vec{x} = \vec{q}_x[k-1]$ e $\vec{u} = \vec{u}[k]$. A matriz H_x, apresentada na equação 3.12 corresponde à jacobiana de $h(\vec{x})$ avaliada em $\vec{x} = \vec{q}_x[k]$.

$$F_x[k] = \left.\frac{\partial f(\vec{x}, \vec{u})}{\partial \vec{x}}\right|_{\vec{x}=\vec{q}_x[k-1], \vec{u}=\vec{u}[k]} = \cos(\frac{1}{2}|\vec{u}[k]|\,\Delta t)\mathrm{I}_{4x4} + \frac{\sin(\frac{1}{2}|\vec{u}[k]|\,\Delta t)}{|\vec{u}[k]|\,\Delta t} W[k]\Delta t \tag{3.10}$$

$$F_u[k] = \left.\frac{\partial f(\vec{x},\vec{u})}{\partial \vec{u}}\right|_{\vec{x}=\vec{q}_x[k-1],\vec{u}=\vec{u}[k]} =$$
$$- \frac{\Delta t}{2 * |\vec{u}[k]|} \sin\left(\frac{1}{2}|\vec{u}[k]|\,\Delta t\right) \mathrm{I}_{4x4}\vec{q}_x[k-1]\vec{u}^T[k]$$
$$+ \left[\frac{2\Delta t}{|\vec{u}[k]|^2}\cos\left(\frac{1}{2}|\vec{u}[k]|\,\Delta t\right) - \frac{4}{|\vec{u}[k]|^3}\sin\left(\frac{1}{2}|\vec{u}[k]|\,\Delta t\right)\right] W[k]\vec{q}_x[k-1]\vec{u}^T[k]$$
$$+ \frac{\sin\left(\frac{1}{2}|\vec{u}[k]|\,\Delta t\right)}{|\vec{u}|[k]} \begin{bmatrix} q_3[k] & -q_2[k] & q_1[k] \\ q_2[k] & q_3[k] & -q_0[k] \\ -q_1[k] & q_0[k] & q_3[k] \\ -q_0[k] & -q_1[k] & -q_3[k] \end{bmatrix} \quad (3.11)$$

$$H_x = \left.\frac{\partial g}{\partial x}\right|_{x=q_x[k]} = \begin{bmatrix} \mathrm{I}_{4x4} \\ \vec{q}_x^T[k]/\,|\vec{q}_x[k]| \end{bmatrix} \quad (3.12)$$

O vetor de comparação calculado pelo método TRIAD não depende das medidas de velocidade angular que compõem o vetor de controle, portanto, a jacabiana G_u do vetor de comparação em função do vetor de controle é nula. A atitude calculada pelo algoritmo TRIAD é função do vetor de medidas formado pelo vetor de medidas do acelerômetro $\vec{a}$ e o vetor de medidas do magnetômetro $\vec{m}$, conseqüentemente o vetor de comparação é função do vetor de medidas $\vec{s} = [\vec{a}\ \vec{m}]$. A jacobiana do vetor de comparação em função do vetor de medidas $\vec{s}$ é calculada pela aproximação numérica apresentada na equação 3.13.

$$\frac{\partial g}{\partial s_i} = \frac{g(\vec{s} + \Delta s_i \hat{i}) - g(\vec{s})}{\Delta s_i} \quad (3.13)$$

em que s_i é a i-ésima componente do vetor de medida $\vec{s}$ e $\hat{i}$ é o vetor de base canônica na direção do eixo de medida de s_i.

O Filtro de Kalman Estendido Correlato *FKEC.m* (ANEXO III) segue o algoritmo 3.

Algoritmo 3 - Algoritmo de implementação do FKEC

CONDIÇÃOES INICIAIS

$P[1]$

Q_0

R_0

$\vec{q}_x[1]$

LAÇO PRINCIPAL

iteração inicial em k=2

AQUISIÇÃO DE DADOS

$\vec{a}$ =leitura do aclerômetro

$\vec{g}$ =leitura do giroscópip

$\vec{m}$ =leitura do magnetômetro

CALIBRAÇÃO

$$\vec{a}_c = \begin{bmatrix} \frac{1}{\max(\vec{A}_x)-b_x}(A_{e,x}-b_x) \\ \frac{1}{\max(\vec{A}_y)-b_y}(A_{e,y}-b_y) \\ \frac{1}{\max(\vec{A}_z)-b_z}(A_{e,z}-b_z) \end{bmatrix}$$

$$\vec{g}_c = \frac{1}{14.375}(\vec{g}_e - \vec{g}_g yro)$$

$$\vec{m}_c = M_s(\vec{m}_e - \vec{b}_m)$$

ETAPA DE PREDIÇÃO

$$\vec{q}_x[k] = f(\vec{q}_x[k-1], \vec{u}[k])$$

$$F_x = \left.\frac{\partial f(\vec{x}\vec{u})}{\partial \vec{x}}\right|_{\vec{x}=\vec{q}_x[k-1],\vec{u}=\vec{u}[k]}$$

$$F_u = \left.\frac{\partial f(\vec{x}\vec{u})}{\partial \vec{u}}\right|_{\vec{x}=\vec{q}_x[k-1],\vec{u}=\vec{u}[k]}$$

$$Q[k] = F_u \eta_u F_u^T + Q_0$$

$$P[k] = F_x P[k-1] F_x^T + Q[k]$$

ETAPA DE CORREÇÃO

$$\vec{q}_y = \begin{bmatrix} \vec{q}_{triad}* \\ 1 - |\vec{q}_{triad}| \end{bmatrix}$$

$$H_x = \left.\frac{\partial h(\vec{x})}{\partial \vec{x}}\right|_{\vec{x}=\vec{q}_x[k]}$$

Cálculo numérico de G_s - Equação 3.13

$$R[k] = G_s \eta_s G_s^T + R_0$$

$$K = \left(P[k]H_x^T\right)\left(H_x P[k] H_x^T + R[k]\right)^{-1}$$

$$\vec{q}_x[k] = \vec{q}_x[k] + K\left(\vec{q}_y - H_x \vec{q}_x[k]\right)$$

$$P[k] = (I_{4x4} - KH_x)P[k](I_{4x4} - KH_x)^T + K[k]RK^T[k]$$

*O quatérnio q_{triad} refere-se ao quatérnio calculado pelo algoritmo TRIAD

3.6 MODELAGEM MATEMÁTICA

Um modelo matemático da atitude de um helicóptero convencional em vôo livre não se adequaria bem à plataforma de teste desenvolvida, pois estes modelos não consideram as restrições e efeitos inseridos pela plataforma e o efeito da barra estabilizadora presente no *BELT-CPX*. Vários trabalhos apresentam modelagens de helimodelos de dinâmica simplificada fixado em plataformas, estes modelos não se adequariam a este trabalho pois o helimodelo *BELT CPX* apresenta dinâmica complexa simular à dinâmica de helicópteros convencionais. A modelagem obtida neste trabalho considerou características e soluções apresentadas em vários trabalhos para a obtenção de um modelo novo que melhor representa o sistema desenvolvido. Os resultados desta modelagem foram publicados no artigo científico *MODELAGEM DE UM HELICÓPTERO ELÉTRICO EM UMA PLATAFORMA 3DOF* apresentado no XIX CREEM [30].

A dinâmica do helicóptero é descrita por um modelo não linear com estados correspondentes aos três ângulos e três velocidades angulares da rotação do corpo do helicóptero, dois ângulos de inclinação do plano de rotação das pás do rotor principal e dois ângulos de inclinação da barra estabilizadore do rotor principal e as entradas correspondentes aos comandos coletivo , pedal , cíclico lateral e cíclico longitudinal. O modelo matemático foi obtido da observação das forças e torques que agem sobre o corpo do helicóptero fixo à plataforma e da relação entre essas forças e torques com os sinais de controle e o estado atual do helicóptero.

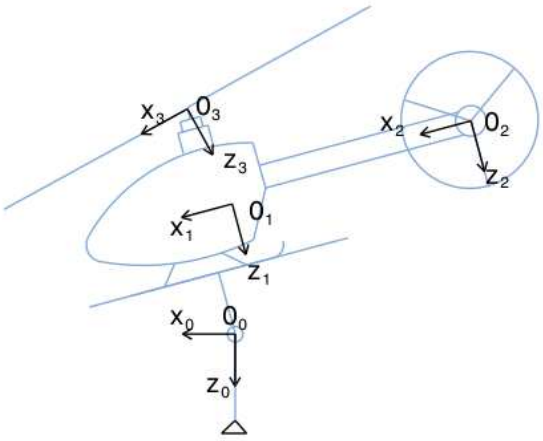


Figura 3.8: Sistemas de coordenadas utilizados na modelagem matemática

O sistemas de coordenadas utilizados na modelagem estão mostrados na figura 3.8. O sistema de coordenada O_0 é uma referência do tipo *n-frame* com origem fixa, localizada no centro de rotação da junta da plataforma, e orientação fixa em relação à Terra. O sistemas O_1 e O_2 são referências do tipo *b-frame*, possuem a mesma orientação fixa em relação ao corpo do helicóptero e origens no centro de massa do helicóptero e no centro de rotação do rotor de cauda, respectivamente. O sistema de coordenadas O_3 possui origem no centro de rotação do rotor principal e orientação fixa em relação ao plano de rotação principal. Quando o helicóptero estiver na posição de referência, correspondente a todos os ângulos de rotação nulos, a orientação dos sistemas O_1 e O_2 será igual à orientação do sistema O_0. E quando o plano de rotação do rotor principal (plano TPP) estiver na orientação de referência (plano HP), a orientação do sistema O_3 será igual à orientação do sistema O_1. A figura 3.9 ilustra os planos TPP e HP. O sistema

O_0 foi escolhido como referência para a representação da atitude por possuir orientação e posição fixas independentes do movimento do helicóptero. Os outros sistemas de coordenadas foram utilizados por serem mais convenientes para alguns cálculos.

3.6.1 Dinâmica do Rotor Principal e a Barra Estabilizadora

Uma maneira de representar a dinâmica de batimento do rotor principal considerando o efeito da barra estabilizadora é a inclusão de quatro estados no sistema, referentes aos ângulos de batimento da hélice(β_{1s} e β_{1c}) e aos ângulos de batimento da barra estabilizadora (β_{1sfly} e β_{1cfly}). Entende-se por ângulo de batimento os ângulos de inclinação do plano formado pela ponta das pás ao girarem no ar, a figura 3.9 apresenta os ângulos de batimento da hélice do rotor principal.

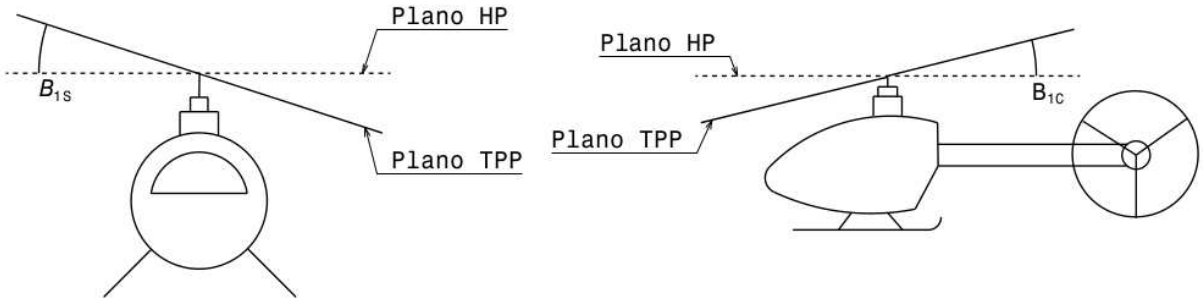


Figura 3.9: Ângulos de batimento do rotor principal

A Equação 3.14 apresenta as expressões simplificadas para a dinâmica do batimento da hélice do rotor principal acoplada à dinâmica de batimento da barra estabilizadora. Os termos de velocidade angular do helimodelo $\dot{\theta}$ e $\dot{\phi}$ estão incluídos para que os ângulos de batimento β_{1c} e β_{1s} representem a orientação do plano de rotação em relação ao sistema de coordenas O_0 [31].

$$\dot{\beta}_{1c} = -\dot{\theta} + \frac{\beta_{1c}}{\zeta_{mr}} + \frac{A_{lon}(U_{lon} + K_c\beta_{1cfly})}{\zeta_{mr}} \tag{3.14a}$$

$$\dot{\beta}_{1s} = -\dot{\phi} + \frac{\beta_{1s}}{\zeta_{mr}} + \frac{B_{lat}(U_{lat} + K_d\beta_{1sfly})}{\zeta_{mr}} \tag{3.14b}$$

$$\dot{\beta}_{1cfly} = -\dot{\phi} + \frac{\beta_{1cfly}}{\zeta_{fly}} + \frac{C_{lon}U_{lon}}{\zeta_{fly}} \tag{3.14c}$$

$$\dot{\beta}_{1sfly} = -\dot{\theta} + \frac{\beta_{1sfly}}{\zeta_{fly}} + \frac{D_{lat}U_{lat}}{\zeta_{fly}} \tag{3.14d}$$

em que ζ_{mr} e ζ_{fly} são as constantes de tempo da hélice e das barras estabilizadoras, respectivamente, A_{lon} e C_{lon} são os ganhos longitudinais e B_{lat} e C_{lat} são os ganhos laterais.

3.6.2 Forças geradas pelos rotores

Uma representação detalhada do processo de geração do empuxo nos rotores leva a uma expressão matemática muito complexa e com muitos parâmetros a serem determinados. Desta forma, para evitar o excesso de parâmetros desconhecidos, optou-se por utilizar uma expressão simplificada apresentada em [32].

3.6.2.1 Rotor principal

O empuxo do rotor principal T_{MR}, módulo da força gerada, é descrito pela equação 3.15.

$$T_{MR} = (w_b - v_i) \cdot \frac{\rho \cdot \Omega \cdot R^2 \cdot a \cdot B \cdot c}{4} \tag{3.15a}$$

$$v_i^2 = \left[\left(\frac{\hat{v}^2}{2} \right)^2 + \left(\frac{T_{MR}}{2 \cdot \rho\pi . R^2} \right)^2 \right]^{0.5} - \frac{\hat{v}^2}{2} \tag{3.15b}$$

$$w_b = \frac{2}{3} \cdot \Omega \cdot R \cdot U_{col} + w_r^2 \tag{3.15c}$$

em que ρ é a densidade do ar, Ω é a velocidade angular do rotor, R é o raio da circunferência formada pelo caminho da ponta das pás em rotação, B é o número de lâminas, c é o comprimento radial da hélice, v_i é a velocidade do vento induzido através do plano TPP, w_b é a velocidade relativa entre as lâminas do rotor principal e o ar, $\hat{v}$ é a velocidade de translação lateral e w_r é a velocidade do rotor em relação ao ar devido às velocidades de translação.

Para o helicóptero fixo à plataforma, sem movimentos de translação, as variáveis $\hat{v}$ e w_r são nulas, e as equações 3.15b e 3.15c podem ser simplificadas para as equações 3.16

$$v_i = \left(\frac{T_{MR}}{2 \cdot \rho \cdot \pi \cdot R^2} \right)^{0.5} \tag{3.16a}$$

$$w_b = \frac{2}{3} \cdot \Omega \cdot R \cdot U_{col} \tag{3.16b}$$

A força gerada pelo rotor principal F_{MR} atua na direção perpendicular ao plano TPP no sentido oposto à fuselagem, no sistema de coordenadas O_3 a fora do rotor principal atua no eixo z com sentido negativo. A equação 3.17 enfatiza a representação da força gerada pelo rotor principal no sistema de coordenadas O_3

$${}^{O_3}\vec{F}_{MR} = \begin{bmatrix} 0 \\ 0 \\ -T_{MR} \end{bmatrix} \tag{3.17}$$

A representação da força gerada pelo rotor principal no sistema de coordenadas O_0 é obtida analisando as suas projeções nos eixos de O_0, que dependem apenas dos ângulos de batimento do rotor principal, de forma que quando o plano de rotação do rotor principal (plano TPP) estiver na orientação de referência

(plano HP) as representações de F_{MR} em O_0 e O_3 são iguais. A Equação 3.18 apresenta a representação da força gerada pelo rotor principal no sistema de coordenadas O_0.

$$^{O_1}\vec{F}_{MR} = -T_{MR} \begin{bmatrix} \sin(\beta_{1c}) \\ -\sin(\beta_{1s}) \\ \cos(\beta_{1c})\cos(\beta_{1s}) \end{bmatrix} \tag{3.18}$$

3.6.2.2 Rotor de cauda

O helimodelo utilizado neste trabalho é equipado com com um sistema de controle de guinada com atuação no rotor de cauda para anular todos os torques que induzem movimentos de guinadas não acionados pelo comando pedal, U_{ped}. O empuxo do rotor de cauda pode ser modelo como a soma de uma parcela que anula os torques de guinada com uma parcela proporcional ao comando U_{ped}.

$$T_{RC} = -\frac{\tau_{T,z^0}}{l_{rc,x}} + U_{ped} \tag{3.19}$$

em que τ_{T,z^0} é o torque total atuando sobre o eixo z do sistema de coordenadas O_0 e l_{rc,x^0} é a projeção no eixo x do sistema de coordenadas O_0 da distância entre o centro do rotor de cauda e o centro de rotação da plataforma.

A força do rotor de cauda atua sempre no eixo y do sistema de coordenadas O_2 no sentido positivo.

$$^{O_2}\vec{F}_{RC} = \begin{bmatrix} 0 \\ T_{RC} \\ 0 \end{bmatrix} \tag{3.20}$$

A representação da força gerada pelo rotor de cauda em O_0 é obtida a partir da representação em O_2 aplicando a matriz de cossenos diretores correspondente à rotação de O_2 em relação à O_0, que equivale à matriz de rotação correspondente aos ângulos de orientação do helimodelo.

$$^{O_0}\vec{F}_{RC} = \begin{bmatrix} F_{RC,x^0} \\ F_{RC,y^0} \\ F_{RC,z^0} \end{bmatrix} \begin{bmatrix} \mathrm{c}(\psi)\mathrm{c}(\theta) & \mathrm{c}(\psi)\mathrm{s}(\phi)\mathrm{s}(\theta) - \mathrm{c}(\phi)\mathrm{s}(\psi) & \mathrm{s}(\phi)\mathrm{s}(\psi) + \mathrm{c}(\phi)\mathrm{c}(\psi)\mathrm{s}(\theta) \\ \mathrm{c}(\theta)\mathrm{s}(\psi) & \mathrm{c}(\phi)\mathrm{c}(\psi) + \mathrm{s}(\phi)\mathrm{s}(\psi)\mathrm{s}(\theta) & \mathrm{c}(\phi)\mathrm{c}(\psi)\mathrm{s}(\theta) - \mathrm{c}(\psi)\mathrm{s}(\phi) \\ -\mathrm{s}(\theta) & \mathrm{c}(\theta)\mathrm{s}(\phi) & \mathrm{c}(\phi)\mathrm{c}(\theta) \end{bmatrix} \begin{bmatrix} 0 \\ T_{RC} \\ 0 \end{bmatrix} \tag{3.21}$$

em que c e s são abreviações para cosseno e seno, respectivamente.

3.6.3 Torques

O modelo obtido considera que o helicóptero fixo à plataforma está sobre efeito dos seguintes torques:

- Torque gerado pelo rotor principal $\vec{\tau}_{MR}$
- Torque gerado pelo rotor de cauda $\vec{\tau}_{RC}$

- Torque de reação ao movimento do rotor principal $\vec{\tau}_d$
- Torque de atrito nas peças móveis da plataforma $\vec{\tau}_{at}$
- Torque de efeito giroscópico do rotor principal $\vec{\tau}_{eg}$
- Torque gerado pela aceleração gravitacional $\vec{\tau}_G$

3.6.3.1 Torque gerado pelo rotor principal

O torque gerado pelo rotor principal é calculado pelo produto vetorial entre a força gerada pelo rotor principal $\vec{F}_{MR}$ e o vetor posição $\vec{l}_{MR}$ do centro do rotor principal em relação ao centro de rotação da articulação da plataforma.

$$ {}^{O_0}\vec{\tau}_{MR} = {}^{O_0}\vec{l}_{MR} \times {}^{O_0}\vec{F}_{MR} = \begin{bmatrix} l_{MR,y^0}F_{MR,z^0} - l_{MR,z^0}F_{MR,y^0} \\ l_{MR,z^0}F_{MR,x^0} - l_{MR,x^0}F_{MR,z^0} \\ l_{MR,x^0}F_{MR,y^0} - l_{MR,y^0}F_{MR,x^0} \end{bmatrix} \tag{3.22} $$

O vetor posição $\vec{l}_{MR}$ no sistema de O_0 é dado pela decomposição do segmento de reta entre o centro de articulação da plataforma e o centro do rotor principal. A equação 3.23 apresenta o resultado obtido analisando a posição do segmento de reta de acordo com a orientação do helimodelo, em que l_{mr} é o módulo da distância entre o centro do rotor principal e o centro de articulação da plataforma e α_{mr} é o ângulo entre o vetor $\vec{l}_{MR}$ e o plano xy do sistema de coordenas O_0 quando o helimodelo está na posição de referência $[\phi\ \theta\ \psi] = [0\ 0\ 0]$.

$$ {}^{O_0}\vec{l}_{MR} = \begin{bmatrix} l_{mr}\cos(\alpha_{mr}+\theta) \\ 0 \\ l_{mr}\cos(\alpha_{mr}+\theta)\sin(\phi) \end{bmatrix} \tag{3.23} $$

3.6.3.2 Torque gerado pelo rotor de cauda

O torque gerado pelo rotor de cauda é calculado pelo produto vetorial entre a força gerada pelo rotor de cauda $\vec{F}_{RC}$ e o vetor posição $\vec{l}_{RC}$ do centro do rotor de cauda em relação ao centro de rotação da articulação da plataforma.

$$ {}^{O_0}\vec{\tau}_{RC} = {}^{O_0}\vec{l}_{RC} \times {}^{O_0}\vec{F}_{RC} = \begin{bmatrix} l_{RC,y^0}F_{RC,z^0} - l_{RC,z^0}F_{RC,y^0} \\ l_{RC,z^0}F_{RC,x^0} - l_{RC,x^0}F_{RC,z^0} \\ l_{RC,x^0}F_{RC,y^0} - l_{RC,y^0}F_{RC,x^0} \end{bmatrix} \tag{3.24} $$

O vetor posição $\vec{l}_{RC}$ no sistema de O_0 é obtido de forma análoga ao vetor $\vec{l}_{MR}$. A equação 3.25 apresenta o resultado obtido, em que l_{rc} é o módulo da distância entre o centro do rotor de cauda e o centro de articulação da plataforma, e α_{mr} é o ângulo entre o vetor $\vec{l}_{MR}$ e o plano xy do sistema de coordenas O_0 quando o helimodelo está na posição de referência $[\phi\ \theta\ \psi] = [0^\circ\ 0^\circ\ 0^\circ]$.

$$ {}^{O_0}\vec{l}_{RC} = \begin{bmatrix} l_{rc}\cos(\alpha_{rc}+\theta) \\ 0 \\ l_{rc}\cos(\alpha_{rc}+\theta)\sin(\phi) \end{bmatrix} \tag{3.25} $$

3.6.3.3 Torque de reação ao movimento do rotor principal

O movimento do rotor principal gera um torque de arrasto aerodinâmico perpendicular ao plano TPP no sentido contrário ao sentido de rotação do rotor principal. Em (Koo et al.,2001) [33] uma expressão simplificada para o torque de arrasto do rotor principal em função do empuxo do rotor principal. O modulo do arrasto é definido pela equação 3.26:

$$ Q_{MR} = (A_{Q,MR} \cdot T_{MR}^{1.5} + B_{Q,MR}) \tag{3.26} $$

em que $A_{Q,MR}$ é o coeficiente de relação entre o empuxo e o arrasto e $B_{Q,MR}$ é o arrasto inicial quando o passo das pás do rotor principal é zero.

O rotor principal gira no sentido horário, com torque no eixo z positivo de O_3. Então o torque de arrasto tem a direção do eixo z de O_3 com sentido negativo, pois induz um movimento no sentido contrário ao rotor principal. A equação 3.27 enfatiza a representação do torque de reação no sistema de coordenadas O_3.

$$ {}^{O_3}\tau_d = \begin{bmatrix} 0 \\ 0 \\ -Q_{MR} \end{bmatrix} \tag{3.27} $$

As projeções de τ_d em O_1 são obtidas por decomposição observando os ângulos de inclinação do rotor principal.

$$ {}^{O_1}\tau_d = -Q_{MR} \begin{bmatrix} \sin(\beta_{1c}) \\ -\sin(\beta 1s) \\ \cos(\beta_{1c})\cos(\beta_{1s}) \end{bmatrix} \tag{3.28} $$

A representação do torque de arrasto em O_0 é obtida a partir da representação em O_1 aplicando a matriz de cossenos diretores correspondente à rotação de O_1 em relação à O_0, que equivale à matriz de rotação correspondente aos ângulos de orientação do helimodelo.

$$ {}^{O_0}\tau_d = \begin{bmatrix} \mathrm{c}(\psi)\mathrm{c}(\theta) & \mathrm{c}(\psi)\mathrm{s}(\phi)\mathrm{s}(\theta) - \mathrm{c}(\phi)\mathrm{s}(\psi) & \mathrm{s}(\phi)\mathrm{s}(\psi) + \mathrm{c}(\phi)\mathrm{c}(\psi)\mathrm{s}(\theta) \\ \mathrm{c}(\theta)\mathrm{s}(\psi) & \mathrm{c}(\phi)\mathrm{c}(\psi) + \mathrm{s}(\phi)\mathrm{s}(\psi)\mathrm{s}(\theta) & \mathrm{c}(\phi)\mathrm{c}(\psi)\mathrm{s}(\theta) - \mathrm{c}(\psi)\mathrm{s}(\phi) \\ -\mathrm{s}(\theta) & \mathrm{c}(\theta)\mathrm{s}(\phi) & \mathrm{c}(\phi)\mathrm{c}(\theta) \end{bmatrix} \begin{bmatrix} -Q_{MR}sin(\beta_{1c}) \\ Q_M MR sin(\beta 1s) \\ -Q_M MR \mathrm{c}(\beta_{1c})\cos(\beta_{1s}) \end{bmatrix} \tag{3.29} $$

3.6.3.4 Torque gerado pela gravidade

O torque gerado pela gravidade, apresentado na equação 3.32, é calculado pelo produto vetorial entre a força peso e o vetor posição do centro de gravidade do helicóptero em relação ao centro de rotação da plataforma. A direção da força peso F_G é sempre coincidente com o eixo z positivo do sistema de coordenadas O_0.

$$ {}^{O_0}\vec{F}_G = \begin{bmatrix} 0 \\ 0 \\ m{\cdot}g \end{bmatrix} \tag{3.30} $$

em que m é a massa do helicóptero e g é a aceleração da gravidade.

O vetor posição do centro de gravidade em relação ao centro de rotação pode ser obtido por decomposição do segmento de reta entre esses dois pontos nos eixos coordenadas de O_0. A equação 3.31 apresenta o resultado obtido, em que l_{cg} é o módulo do segmento de reta e α_{cg} é o ângulo entre o segmento de reta e o plano xy de O_0 quando o helicóptero está na orientação de referência $[\phi\ \theta\ \psi] = [0°\ 0\circ\ 0°]$.

$$ {}^{O_1}\vec{l}_{CG} = \begin{bmatrix} l_{cg}\cos\left(\alpha_{cg}+\theta\right) \\ 0 \\ l_{cg}\sin(alpha_{cg}+\theta)\sin\left(\phi\right) \end{bmatrix} \tag{3.31} $$

$$ {}^{O_0}\vec{\tau}_G = {}^{O_0}\vec{l}_{CG} \times {}^{O_0}\vec{F}_G = \begin{bmatrix} m{\cdot}g{\cdot}l_{cg}\cdot\cos\left(\alpha_{cg}+\theta\right)\sin\left(\phi\right) \\ -m{\cdot}g{\cdot}l_{cg}\cdot\cos\left(\alpha_{cg}+\theta\right) \\ 0 \end{bmatrix} \tag{3.32} $$

O torque de atrito é composto por uma parcela de atrito de elástico constante e uma parcela de atrito viscoso proporcional à velocidade de angular.

$$ {}^{O_0}\tau_{at} = \begin{bmatrix} k_{ar}+\mu_{ar}\dot{\phi} \\ k_{aa}+\mu_{aa}\dot{\theta} \\ k_{ag}+\mu_{ag}\dot{\psi} \end{bmatrix} \tag{3.33} $$

em que k_{ar}, k_{aa} e k_{ag} as componentes de atrito elástico de rolagem e μ_{ar}, μ_{aa} e μ_{ag} são os coeficientes de atrito viscoso.

3.6.3.5 Torque de efeito giroscópico

O torque de efeito giroscópico surge por reação a mudanças na orientação do plano de rotação do rotor principal. Ao se aplicar um torque de mudança da orientação de um plano de rotação, observa-se o surgimento de um torque defasado em $90°$ em relação ao torque aplicado e mesma magnitude do torque aplicado. Considerando, por simplificação, que o plano TPP não se move em relação a fuselagem do helicóptero, todo movimento do plano TPP será devido à movimentos do helimodelo. Assim, um movimento de arfagem terá como efeito o surgimento de um torque de rolagem e um movimento de rolagem terá como

efeito o surgimento de um torque de arfagem. Movimentos guinada não causarão efeitos giroscópicos pois não alterarão a orientação do plano TPP. A Equação 3.35 apresenta o torque de efeito giroscópico representado no sistema de coordenadas O_1.

$$ {}^{O_1}\vec{\tau}_{EG} = \begin{bmatrix} I_r\ddot{\theta} \\ I_r\ddot{\phi} \\ 0 \end{bmatrix} \tag{3.34} $$

em que I_r é o momento de inércia do helimodelo fixo à plataforma.

3.6.3.6 Dinâmica de corpo rígido

O torque de efeito giroscópico pode ser representado em O_0 aplicando-se a matriz de cossenos diretores correspondente aos ângulos de orientação do helimodelo à representação de $\vec{\tau}_{EG}$ em O_1.

$$ {}^{O_0}\vec{\tau}_{EG} = \begin{bmatrix} \mathrm{c}(\psi)\mathrm{c}(\theta) & \mathrm{c}(\psi)\mathrm{s}(\phi)\mathrm{s}(\theta) - \mathrm{c}(\phi)\mathrm{s}(\psi) & \mathrm{s}(\phi)\mathrm{s}(\psi) + \mathrm{c}(\phi)\mathrm{c}(\psi)\mathrm{s}(\theta) \\ \mathrm{c}(\theta)\mathrm{s}(\psi) & \mathrm{c}(\phi)\mathrm{c}(\psi) + \mathrm{s}(\phi)\mathrm{s}(\psi)\mathrm{s}(\theta) & \mathrm{c}(\phi)\mathrm{c}(\psi)\mathrm{s}(\theta) - \mathrm{c}(\psi)\mathrm{s}(\phi) \\ -\mathrm{s}(\theta) & \mathrm{c}(\theta)\mathrm{s}(\phi) & \mathrm{c}(\phi)\mathrm{c}(\theta) \end{bmatrix} \begin{bmatrix} I_r\ddot{\theta} \\ I_r\ddot{\phi} \\ 0 \end{bmatrix} \tag{3.35} $$

Aplicando-se a segunda lei de Newton para rotações ao torque resultante obtém-se um modelo não linear que rege a dinâmica do helimodelo acoplado à plataforma de testes. A equação 3.36 apresenta a relação entre as acelerações angulares e o torque total que é função das variáveis de estado: orientação, velocidade angular, ângulos de batimento da hélice do rotor principal e ângulos de batimento do rotor principal.

$$ {}^{O_0}\vec{\tau}_T = \begin{bmatrix} \tau_{T,x^0} \\ \tau_{T,y^0} \\ \tau_{T,x^0} \end{bmatrix} = {}^{O_0}\vec{\tau}_{MR} + {}^{O_0}\vec{\tau}_{RC} + {}^{O_0}\vec{\tau}_d + {}^{O_0}\vec{\tau}_g + {}^{O_0}\vec{\tau}_{at} + {}^{O_0}\vec{\tau}_G + {}^{O_0}\vec{\tau}_{EG} \tag{3.36} $$

$$ \ddot{\phi} = \frac{\tau_{T,x^0}}{I_r} \tag{3.37a} $$

$$ \ddot{\theta} = \frac{\tau_{T,y^0}}{I_r} \tag{3.37b} $$

$$ \ddot{\psi} = \frac{\tau_{T,z^0}}{I_r} \tag{3.37c} $$

4 RESULTADOS

"A baby has brains, but it doesn't know much. Experience is the only thing that brings knowledge, and the longer you are on earth the more experience you are sure to get"

- L. Frank Baum, The Wonderful Wizard of Oz

4.1 INTRODUÇÃO

A plataforma desenvolvida foi avaliada em teste de desempenho da interface de aquisição de dados e atuação e em teste de coerência da estimativa de atitude fornecida pelo sistema de navegação inercial. O modelo matemático obtido para a dinâmica do helicóptero fixo à plataforma foi transcrevido para *Matlab-Simulink* e simulações foram realizadas para validar a coerência da modelagem. O modelo computacional também foi utilizado para simulação de um sistema de controle da atitude do helimodelo com controladores PID discretos, a intenção dessa simulação foi demonstrar que a estimativa de atitude fornecida pela plataforma é suficiente como sinal de realimentação do sistema de controle da orientação do helimodelo e que controladores simples são capazes de fazer com que a atitude do helimodelo fixo à plataforma siga sinais de referência.

4.2 DESEMPENHO DA INTERFACE DE AQUISIÇÃO DE DADOS E ATUAÇÃO

4.2.1 Objetivo

O objetivo do teste de desempenho da interface é determinar se a taxa de 50Hz para leitura dos sensores e atualização do sinal de acionamento dos motores é suportada pela interface, e registrar o atraso em número de períodos entre a leitura dos sensores e o respectivo acionamento dos motores, a maior diferença entre o instante em que o sensor atualiza sua saída e o instante de leitura da saída do sensor, e o atraso na atualização dos sinais de acionamento dos motores.

A taxa de amostragem da interface de aquisição de dados é limitada pelo tempo que o sistema demora para ler todos os sensores, enviar os dados lidos para computador e atualizar o sinal PWM de acionamento dos cinco motores. E existe um atraso entre a leitura dos sensores e a disponibilização dos sinais de comando pelo computador que pode fazer com que os dados recebidos em um instante de amostragem não influênciem no acionamento seguinte. Para que não ocorra perda de dados enviados da interface para o computador, o atraso entre a leitura do sensor e o respectivo acionamento do motor deve ser inferior ao período de amostragem.

Os estados dos motores são atualizados atualizando-se o sinal PWM de controle de cada motor. Como essa tarefa é realizada de forma serial existe um atraso entre o instante de amostragem desejado que é igual

para todos os motores e atualização do PWM enviado para cada motor.

A placa *SEN-10724* permite apenas a leitura de um registrador de 8 bits por consulta pelo barramento I^2C e cada um dos seus três sensores apresenta três saídas de 16 bits para serem lidas. Assim, a leitura completa exige um total de 18 comandos de leitura pelo barramento de I^2C e existe um atraso de leitura entre o instante de leitura desejado e o final da leitura do sensor. O giroscópio, acelerômetro e magnetômetro utilizados atualizam suas saídas a taxas de 1KHz, 1.6KHz e 75 Hz, respectivamente,que são superiores aos 50Hz de taxa de leitura desejada. Assumindo que apenas a dessincronização entre o instante de leitura da interface e o instante de atualização da saída do sensor e o tempo de leitura de sensor são fontes de atraso, a defasagem máxima entre o instante de atualização e o instante de leitura do sensor é igual ao atraso de leitura se este for maior do a que período de atualização do sensor, caso contrário, é igual a diferença entre a período de atualização do sensor e o atraso de leitura em relação ao instante de leitura desejado. O cálculo da defasagem máxima $\Delta\tilde{T}$ está explicitado na equação 4.1, em que $\Delta\tilde{T}_r$ é o atraso de leitura do sensor e f_{si} é a frequência de amostragem do sensor.

$$\Delta\tilde{T} = \begin{cases} \Delta\tilde{T}_r & se \Delta\tilde{T}_r \geq \frac{1}{2f_{si}} \\ \frac{1}{2f_{si}} - \Delta\tilde{T}_r & se \Delta\tilde{T}_r < \frac{1}{2f_{si}} \end{cases} \tag{4.1}$$

A duas versões de interface foram testadas. Como critério de aprovação espera-se que a interface opere com o taxa de amostragem de 50Hz. Os dados sem relação direta com período de amostragem foram levantados para carcterização da plataforma e disponibilização para consulta futura.

4.2.2 Interface baseada em Arduino

A implementação em Arduino divide as tarefas da interface em dois processos, atuação e aquisição, que são executados serialmente intercalados e sincronizados pelo *Timer4*. A informação do tempo de execução de um conjunto de comandos é extraída do registrador TCNT4 que armazena o valor de contagem do *Timer4* e é zerado no início de cada um dos processos. A partir da diferença entre o valor armazenado em TCNT4, antes e depois de um conjunto, o tempo de execução desse conjunto de comandos pode ser obtida pela equação 4.2, em que k é a constante de divisão de clock configurada e ΔC é a diferença do valor armazenado no registrador TCNT4 entre os dois insttantes de tempo.

$$\Delta\tilde{T}_r = k \cdot \Delta C \cdot \frac{1}{16000000} \tag{4.2}$$

4.2.2.1 ATRASOS NA LEITURA E ATUAÇÃO

Os atrasos nas leituras dos sensores são medidos em relação ao início da iteração do processo de aquisição, e os atrasos nas atualizações dos sinais de acionamento dos motores são medidos em relação ao início da iteração do processo de atuação . Como no início de cada iteração o registrador TCNT4 é zerado, os atrasos de leitura dos sensores e de atualização do acionamento dos motores são calculados pela equação 4.2, com ΔC igual ao valor registrado em TCNT4 após a leitura dos sensores e após a atualização do acionamento dos motores, respectivamente.

Tabela 4.1: Tempos de desempenho da plataforma em Arduino

Tempo máximo entre atualização e leitura acelerômetro	228μs
Tempo máximo entre atualização e leitura giroscópi	544μs
Tempo máximo entre atualização e leitura do magnetõmetro	12,5ms
Tempo de execução do processo de aquisição	1,12ms
Atraso máximo no acionamento do rotor principal	11μms
Atraso máximo no acionamento do servo de controle do ângulo de batimento lateral do rotor principal	20μs
Atraso máximo no acionamento do servo de controle do ângulo de batimento longitudinal do rotor principal	32μs
Atraso máximo no acionamento do servo de controle do ângulo de batimento do rotor cauda	43μs
Tempo máximo de execução do processo de atuação	43μs

O Arduino foi programado para a cada iteração do processo de aquisição enviar para o computador o valor registrado em TCNT4 após a leitura do acelerômetro. Um programa em *Matlab* foi utilizado para calcular o valor máximo de 5000 amostras de TCNT4 e convertê-la para segundos pela equação 4.2. A partir do resultado obtido a diferença máxima entre o instante de atualização e o instante de leitura do acelerômetro foi calculada pela 4.1. O mesmo procedimento foi seguido para obter a diferença máxima entre a atualização e leitura do giroscópio e do magnetômetro. Para determinar o tempo de execução de uma iteração do processo de aquisição o mesmo procedimento foi seguido com o Arduino enviando o valor de TCNT4 no final da iteração do processo de aquisição, após o envio das leituras dos sensores para o computador. A mesma metodologia foi aplicada ao processo de atuação para estimar o atraso de acionamento de cada um dos motores e o tempo de execução de uma iteração do processo de atuação. A Tabela 4.1 apresenta os resultados obtidos.

O menor período de amostragem possível de ser realizado com a interface implementada em Arduino é 1.2ms, dobro do maior tempo de execução dos processos, o que corresponde a uma taxa máxima de 833Hz, superior aos 50Hz estipulados como critério de aprovação, portanto o desenho da interface implemntada em Arduino é satisfatório. É importantante lembrar que mesmo que o Arduino seja capaz de realizar leituras dos sensores a cada 1.2ms, o magnetômetro possui uma taxa de atualização máxima de 75Hz.

4.2.2.2 ATRASOS ENTRE E LEITURA E ATUAÇÃO

A leitura e o acionamento dos motores, na versão de interface baseada em Arduino, não ocorrem paralelamente, existe um atraso de meio período de amostragem. Para que as leituras dos sensores em um instante de amostragem influenciem na próxima atuação é necessário que o tempo total entre o começo da leitura dos sensores e a disponibilização dos sinais de comando pelo computador seja menor do que este atraso.

O processo de aquisição foi programado para armazenar na variável 'N1' o número de iterações já realizadas, e enviar para o computador, junto com a leitura dos sensores, o valor de 'N1'. A aplicação

no computador, responsável por estimar a atitude e calcular o sinal de comando, foi programada para armazenar em ’N2’ o valor de ’N1’ enviada pelo processo de aquisição, e enviar para o Arduino o valor de ’N2’ junto com o sinal de comando. O processo de atuação foi programado para armazenar na variável ’N3’ o valor de ’N2’, e se ’N3’ for igual a ’N1’, condição que será satisfeita se o tempo decorrido da leitura dos sensores até a obtenção do comando for menor do que o atraso entre os processos, enviar ’OK’ para o computador, e se ’N3’ for menor do que ’N1’, condição que será satisfeita se o tempo até um valor de ’N1’ passar para ’N2’ e chegar em ’N3’ for suficiente para que o processo de aquisição repita e incremente ’N1’, enviar para o computador a diferença entre ’N3’ e ’N1’ que corresponde ao atraso de períodos em amostragem entre a leitura e o acionamento dos motores. Por exemplo, se o período de amostragem for 10ms, o atraso entre uma iteração de aquisição e uma iteração de atuação será 5ms, na primeira iteração o valor de ’N1’ será ’1’, se em menos de 5ms as leituras forem enviadas para o computador e o computador calcular e enviar os comandos, quando o processo de atuação for executado o valor de ’N2’ recebido será igual a ’N1=1’ e o Arduino enviará ’OK’ para o computador, ou, se o tempo da leitura dos sensores até o computador enviar os comandos for igual 10ms, quando o Arduino ler o valor de ’N2’ o processo de atuação já terá sido executado mais uma vezes desde o envio de ’N1=1’ e o valor de ’N1’ será igual a 2 quando o valor de ’N2=1’ for recebido em ’N3’, com ’N1=2’ e ’N3=1’, o valor enviado para o computador será 1 que corresponde ao atraso de um período de amostragem entre a leitura e o acionamento dos motores. Lembrando que existe um atraso adicional de meio período de amostragem devido.

Observou-se da execução do teste com taxa de amostragem de 50Hz que a interface envia para o computador ’OK’, o que indica que cada leitura do sensor é capaz de influenciar na atualização seguinte do sinal PWM enviado para os motores e que não ocorre perda de dados.

4.2.3 Interface baseada em Raspberry Pi

A interface implementada em *Raspberry Pi* divide as tarefas em três *tasks*: leitura, recebimento e atuação, que são executadas paralelamente e sincronizadas por alarmes. A primeira função do laço do principal das *tasks* leitura e atuação é *rt_alarm_wait(ALARME_1)* que faz com que as *tasks* esperem por um sinal do *ALARME_1* para continuar suas execuções. A leitura dos sensores e a atualização do acionamento dos motores são iniciadas juntas no instante de ocorrência do sinal de *ALARME_1*.

O intervalo entre a ocorrência do sinal de alarme até a conclusão de um comando pode ser obtido da diferença entre o tempo após o comando e o tempo lido após *rt_alarm_wait(ALARME_1)*. A biblioteca *timer.h*, descrita em [34], disponibiliza a função *rt_timer_read()* que retorna o tempo no instante em que é executada.

4.2.3.1 ATRASOS NA LEITURA E ATUAÇÃO

A atraso na medida do acelerômetro foi calculado pela diferença entre valor retornado pela função *rt_timer_read* no ínicio da iteração da *task* leitura e o valor retornado após o final da leitura do acelerômetro, o máximo atraso foi estimado como o máximo valor calculado em 5000 iterações. Em seguida o mesmo procedimento foi seguido para a estimação do atraso na leitura dos outros sensores. Para estimação do tempo de execução da *task* leitura foi calculado o máxmimo tempo decorrido entre a ocorrência do

Tabela 4.2: Tempos de desempenho da plataforma em Raspberry

Erro máximo entre a atualização e leitura do acelerômetro	3,06μs
Erro máximo entre a atualização e leitura do giroscópio	5,53μms
Erro máximo entre a atualização e a leitura do magnetômetro	8,67μs
Tempo máximo de execução da *task* leitura	8,95μs
Atraso máximo no acionamento do rotor principal	1,1μs
Atraso máximo no acionamento do servo de controle do ângulo de batimento lateral do rotor principal	5,89μs
Atraso máximo no acionamento do servo de controle do ângulo de batimento longitudinal do rotor principal	9,02μs
Atraso máximo no acionamento do servo de controle do ângulo de batimento do rotor cauda	12,02μs
Tempo máximo de execução do processo de atuação	12,12μs

alarme e o final do envio das leituras dos sensores para o computador. A mesma metodologia foi aplicada à *task* atuação para a estimação do tempo de atraso na atualização do sinal de acionamento dos motores, como o ultimo comando dessa *task* é o acionamento do motor de cauda, tempo de execução da *task* atuação é igual ao tempo de atraso no acionamento do motor de cauda. A tabela 4.2 apresenta os atrasos máximos estimados.

O menor período de amostragem é definido pelo dobro do maior tempo de execução entre as *tasks* leitura e acionamento. Dos resultados apresnetados na tablea 4.2 observa-se que o maior frequência de amostragem é 41.2MHz, valor muito superior aos 50Hz definidos como critério de aprovação. Portanto, a interface implementada em Raspberry Pi possui desemenpenho satisfatório.

4.2.3.2 ATRASO ENTRE LEITURA E ATUAÇÃO

As ações de leitura dos sensores e atualização dos sinas PWM de acionamento dos motores são executadas ao mesmo tempo, como é impossível que o calculo dos sinais de comando em função das leituras sejam executadas sem atraso, é impossível que a leitura dos sensores no instante k influencie na atuação no instante k. Para evitar o atraso no acionamento devido ao tempo gasto para ler o sinal enviado pelo computador, a *task* recebimento foi criada para realizar a leitura, de forma que após a ocorrência do sinal do *ALARME_1* o primeiro comando da *task* atuação é o acionamento do rotor principal. A *task* é defasada das *tasks* leitura e atuação em meio período de amostragem, assim para que os dados lidos em um instante de tempo k influencie na atuação no instante de tempo $k+1$ é necessário que o tempo decorrido da leitura dos sensores até a dinsponibilização do comando pelo computador seja menor que meio período de amostragem, caso contrário existirá um atraso maior do que um período de amostragem entre a leitura e atuação. Para que não ocorra perda de dados enviados da interface para o computador, o atraso na disponibilização dos sinais de comando deve ser inferior ao período de amostragem.

Para identificar se as medidas feitas em um instante de amostragem podem influenciar no acionamento dos motores no instante de amostragem seguinte foi seguida a mesma metodologia adotada para interface baseada em Arduino. A *task* leitura foi programada para registrar na variável 'N1' o número de repetições já realizadas da thread leitura e enviar o valor de 'N1' junto com a leitura dos sensores. A aplicação em *Matlab*, responsável por estimar a atitude e disponibilizar os comandos de acionamento dos motores, foi programada para armazenar em 'N2' o valor de 'N1' enviado pela interface e enviá-lo para a Raspberry junto com os comandos de acionamento dos motores. A *Task* recebimento foi programada para armazenar em 'N3' o valor de 'N2' enviado pelo computador e compará-lo com 'N1'. Se o valor de 'N3' for igual a 'N1' a *string* 'OK' é impresso no console do terminal, caso contrário a diferença entre 'N3' e 'N1' é impressa no console do terminal. A diferença entre 'N3' e 'N1' corresponde ao atraso em número de períodos de amostragem entre a leitura dos sensores e a disponibilziação pelo computador do comando de acionamento dos motores, se este valor for igual 0, sabe-se que o atraso total é menor do que o período de amostragem e não ocorre perda de dados.

Observou-se da execução do teste a impressão de 'OK' no console terminal, o que indica que sinal de comando de acionamento dos motors no instante k pode ser função da estimativa da atitude no instante $k-1$ ou anterior, e que o atraso não causa perda de dados na comunicação entre o computador e a interface.

4.3 PARÂMETROS DE CALIBRAÇÃO ESTIMADOS

Essa seção apresenta o resultado para um realização dos experimentos realizados para estimar os parâmetros de correção dos erros na saída dos sensores, conforme apresentados na seção 3.3.

ACELERÔMETRO

A Tabela 4.3 apresenta o vetor de viés e os fatores correção de escala obtidos para ADXL345.

Tabela 4.3: Constantes de calibração obtidas para o acelerômetro

Eixo	Viés	Fator de correção de escala
X	-5.5	0.0034
Y	-39.0	0.003
Z	-9.5	0.0038

GIROSCÓPIO

A Tabela 4.3 apresenta o viés obtido para cada um dos eixos de medida do ITG-3200.

Tabela 4.4: Viés estimado para as leituras do giroscópio

Eixo	Viés
X	-22.0700
Y	44.9130
Z	-4.1950

Tabela 4.5: Viés estimado para medidas do magnetômetro

Eixo	Viés
X	77.3798
Y	161.385
Z	-144.782

MAGNETÔMETRO

A tabela 4.3 apresenta o viés obtido para cada eixo de medida do HMC5843.

$$M_s = \begin{bmatrix} 0.83181 & 0.00763019 & -0.0193070 \\ 0.00763019 & 0.884438 & -0.0228394 \\ -0.0193070 & -0.0228394 & 0.992902 \end{bmatrix} \tag{4.3}$$

4.4 ESTIMATIVAS DA ATITUDE

O objetivo deste teste é verificar a coerência das estimativas fornecidas pelo sistema de navegação inercial. Infelizmente até o momento de apresentação deste trabalho não foi possível realizar um experimento para estimar a precisão das estimavas, devido à falta de um instrumento preciso que forneça medidas de orientação para comparação.

O primeiro experimento foi realizado com movimentos de teste aplicados diretamente sobre placa de sensores *SEN-10724*. O segundo experimento foi realizado com a *SEN-10724* fixada na plataforma sobre o palco. O objetivo do primeiro experimento foi testar o sistema de navegação inercial, e objetivo do segundo experimento foi verificar o efeito dos materiais da plataforma nas estimativas, principalmente devido à distorção nas medidas do magnetômetro causada por materiais ferromanético.

O procedimento de teste seguido para os dois experimentos está descrito no proceidmento 1. A figura 4.1 ilustra as orientações da placa *SEN-10724* durante os passos do procedimento 1. E a figura 4.2 apresentam os eixos x, y e z, fixos em relação à placa, citados durante o procedimento 1, os sentidos das rotações seguem a convenção da regra da mão direita.

Procedimento 1	
1	Iniciar o teste com placa imóvel na horizontal, orientação ilustrada na figura 4.1a. A orientação que a placa estiver no começo do teste será a orientação considerada pelo SNS como $\phi = \theta = \psi = 0°$.
2	Realizar um movimento de $+90°$ de arfagem, rotação em torno do eixo *y*. A orientação final está ilustrada na Figura 4.1b.
3	Realizar uma rotação de $-90°$ de arfagem para que a orientação final seja igual à orientação inicial no passo 1
4	Realizar uma rotação de $-90°$ de guinada, rotação em torno do eixo *z*. A orientação final está ilustrada na Fiugra 4.1c.
5	Realizar uma rotação de $+90°$ de guinada para que a orientação final seja igual à orientação inicial no passo 1.
6	Reallizar uma rotação de $-90°$ de rolagem, rotação em torno do eixo *x*. A orientação final está ilustrada na figura 4.1d.
7	Realizar uma rotação de $+90°$ de rolagem para que a orientação final seja igual à orientação inicial no passo 1.

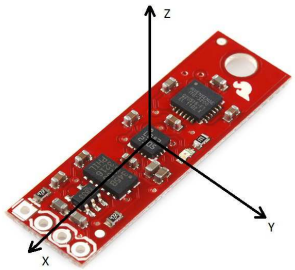


Figura 4.1: Eixos de rotação citados no procedimento de teste da estimativa de atitude

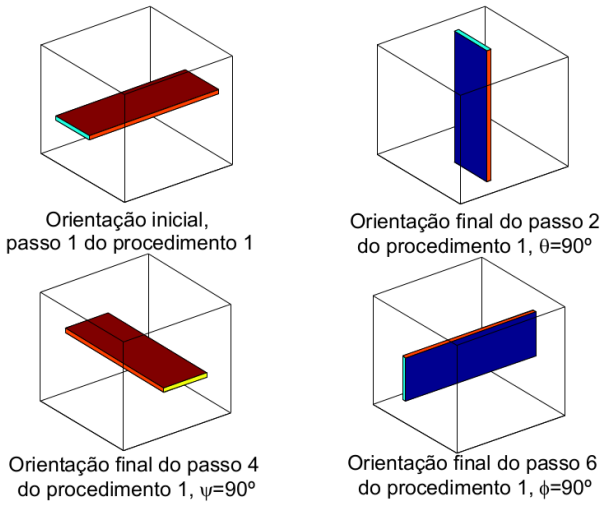


Figura 4.2: Orientações da placa de sensores durante os passos do procedimento 1

A figura 4.3 apresenta as estimativas obtidas apenas pela integração das medidas do giroscópio, assumindo que a posição inicial é $\theta = \phi = \psi = 0°$. Observa-se que no intervalo entre os movimentos a orientação estimada não é aproximadamente igual à orientação inicial, e que essa diferença cresce como tempo, o que ocorre por efeito da propagação temporal do erro pela operação de integração.

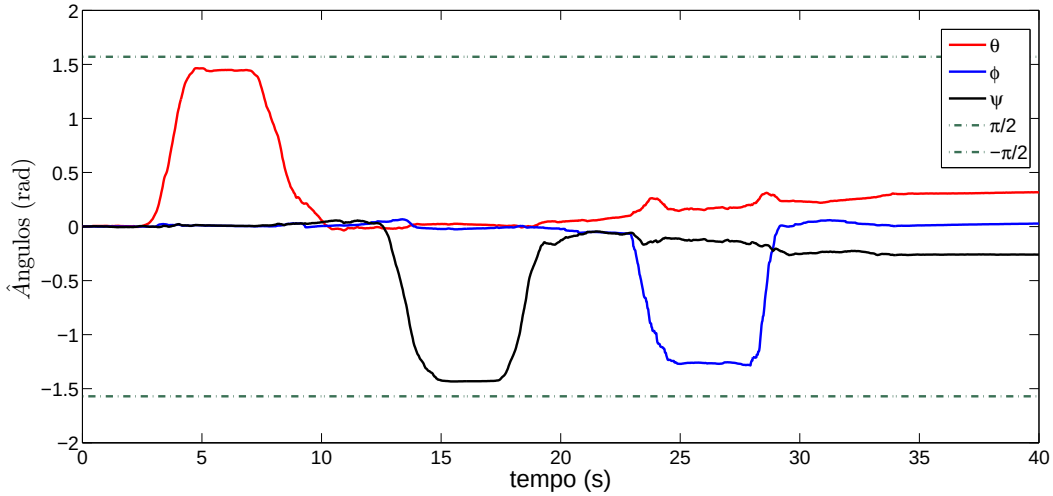


Figura 4.3: Estimativas obtidas da realização do procedimento 1, pela integração das medidas do giroscópio sem o FKEC

A figura 4.4 apresenta as estimativas obtidas apenas pelo método TRIAD tomando a orientação inicial como $\theta = \phi = \psi = 0°$. Observa-se que na estimative TRIAD não ocorre propagação temporal do erro como ocorre na integração das medidas do giroscópio, entretanto observa-se um ruído maior.

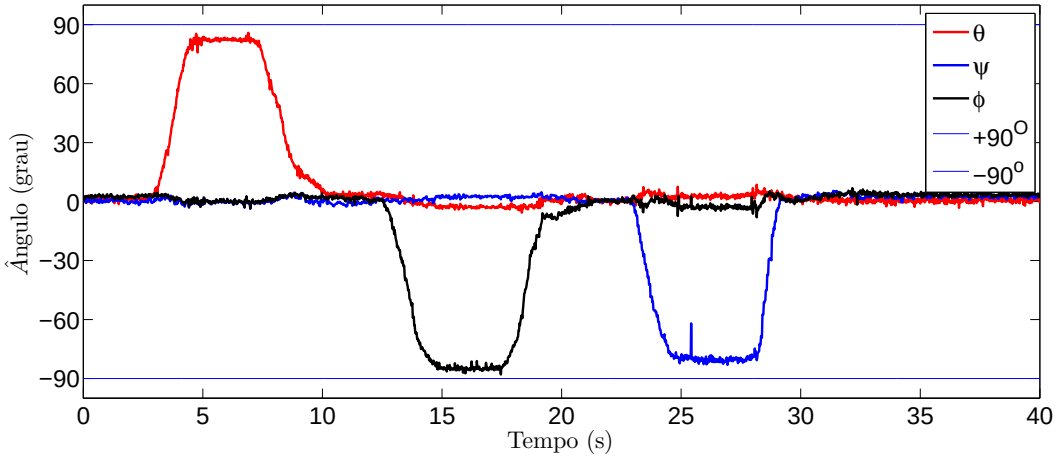


Figura 4.4: Estimativas obtidas da realização do procedimento 1, pelo método TRIAD

As estimativas obtidas pelo sistema de detecção da atitude proposto em teste realizado diretamente sobre a placa de sensores, conforme o procedimento 1, estão apresentadas na figura 4.5. Observa-se que a ordem e a direção das variações dos ângulos estimados é coerente em relação a ordem e direção dos movimentos realizados no experimento, e que após os movimentos a estimativa da orientação permanece

constante praticamente igual à orientação inicial. Comparando as estimativas obtidas pelo SDA proposto com as estimativas obtidas apenas pelo giroscópio ou apenas pelo método TRIAD, observa-se que as estimativas do SDA proposto apresenta ruído inferior as estimativas do método TRIAD sem a propagação temporal no erro observada para as estimativas obtidas por integração das medidas do giroscópio.

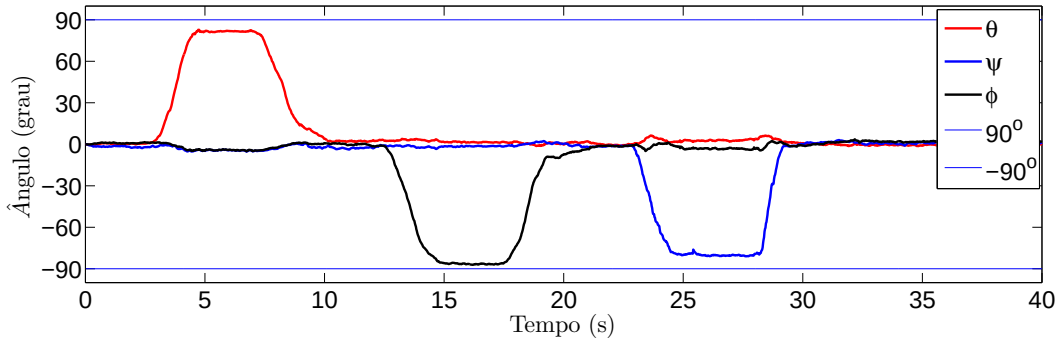


Figura 4.5: Estimativas obtidas durante o teste realizado diretamente sobre a *SEN-10724*, conforme os passos do procedimento 1

Assumindo que os movimentos realizados manualmente foram perfeitamente iguais a $90°$, o maior erro percentual ,entre os valores de pico estimados e os valores de pico esperados, observado no experimento foi de $6,7\%$.

A comparação das estimativas sem o filtro de Kalman com as estimativas filtradas, mostra que o filtro de Kalman implementado é capaz de combinar as daus estimativas para obter uma estimativa sem propagação significativa do erro no tempo e com menos ruído do que a estimativa TRIAD.

O experimento realizado sobre a plataforma apresentou estimativas muito distorcidas, quase sem relação com os movimentos realizados. A fonte de erro identificada foi a junta Cardan de aço utilizada na plataforma em uma posição muito próxima dos sensores, que distorce as medidas do magnetômetro prejudicando as estiamtivas do algoritmo TRIAD. Esse resultado motivou o desenvolvimento da junta descrita na seção 3.2 fabricada de materiais predominantemente não ferromagnéticos.

4.5 SIMULAÇÃO DO MODELO NÃO LINEAR

O modelo matemático levantado para a dinâmica do helicóptero fixado à plataforma, que considera os efeitos de atrito e restrições de movimentos introduzidos pela plataforma e o efeito da barra estabilizadora adicionada a helimodelos, foi implementado em *Matlab-Simulin* para simulações da resposta da orientação do helimodelo aos sinais de comando. Simulações foram realizadas para validar a coerência do modelo.

O *Simulink* foi configurado para utilizar o método Runge-Kutta de quarta ordem com passo de inte-

gração de 20ms na solução numérica das equações diferenciais do modelo matemático. Em todos os casos simulados foi assumido como condição inicial todos os ângulos e velocidades angulares do helimodelo iguais a $0°$, e o plano de rotação TPP coincide com o plano HP.

4.5.1 Sinais de controle iguais a zero

O primeiro caso de teste foi o caso em que todos os sinais de comando são nulos e os dois rotores estão parados. Nessa condição apenas a força de atração gravitacional e as forças de atrito atuam no corpo do helimodelo. Na oriengação inicial, a aceleração gera torque apenas em torno do eixo de arfagem, portanto, espera-se que o helimodelo comporte-se como um pêndulo amortecido em torno do eixo de arfagem, com orientação inicial $\phi = 0°$ e ponto de estabilidade em $\theta = -90° - \alpha_{cg}$ sem variações nos outros ângulos, em que α_{cg} é o ângulo inicial entre o eixo x^0 e o segmento de reta entre o centro de articulação da plataforma e o centro de gravidade do helimodelo. A Figrua 4.6 apresenta os gráficos obtidos da simulação dos ângulos de atitude e velocidades angulares.

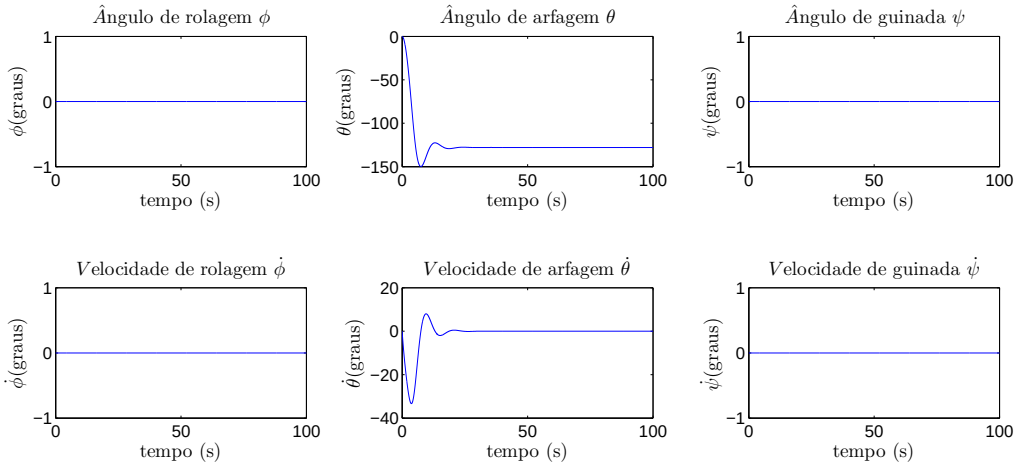


Figura 4.6: Gráficos dos ângulos e velocidades angulares, para sinais de controle nulos

4.5.2 Vôo Pairado

O segundo caso de teste foi o caso em que os comandos satisfazem as condição de vôo pairado no instante t=0 em que o helicóptero e o plano de rotação do seu rotor principal estão na horizontal. Não é possível obter um vôo pairado permanente com comandos constantes, porquê o helicóptero é um sistema instável em malha aberta, por esse motivo o tempo de simulação foi limitado a 3s, pois sem a presença de um controlador as variáveis de interesse divergem rapidamente impedindo a análise.

Para que o helicóptero esteja em vôo pairado na horizontal com o plano de rotação do rotor principal na horizontal é necessário que o torque gerado pelo rotor principal compense o torque gerado pela gravidade para que não haja movimento de arfagem. Como o torque do rotor principal é função do comando coletivo, existe um valor de comando coletivo que promove o equilíbrio entre o torque do rotor principal e o torque

de gravidade. E para que o plano de rotação do rotor principal permaneça parado na horizontal é necessário que os comandos cíclicos sejam nulos.

O movimento do rotor principal induz um movimento de arfagem que é compensado pelo rotor de cauda, entretanto o movimento do rotor de cauda, além de anular o movimento de arfagem induzido pelo rotor principal, induz movimentos de rolagem que não possuem um mecanismos de controle. O movimento de rolagem, faz com que o torque do rotor principal seja decomposto em um torque de rolagem e um torque arfagem insuficiente para compensar o torque da gravidade que induz movimentos de arfagem. Assim, o resultado esperado é que inicialmente o helicóptero mantenha-se aproximadamente parado, com velocidades baixas, e que os ângulos de rolagem e arfagem que começam em zero, afastem-se exponencialmente da orientação inicial, pois o seu aumento faz com que aumentem mais. A Figura 4.7 apresenta os gráficos obtidos para os ângulos de atitude e empuxo do rotor principal, em que observa-se os comportamentos esperados.

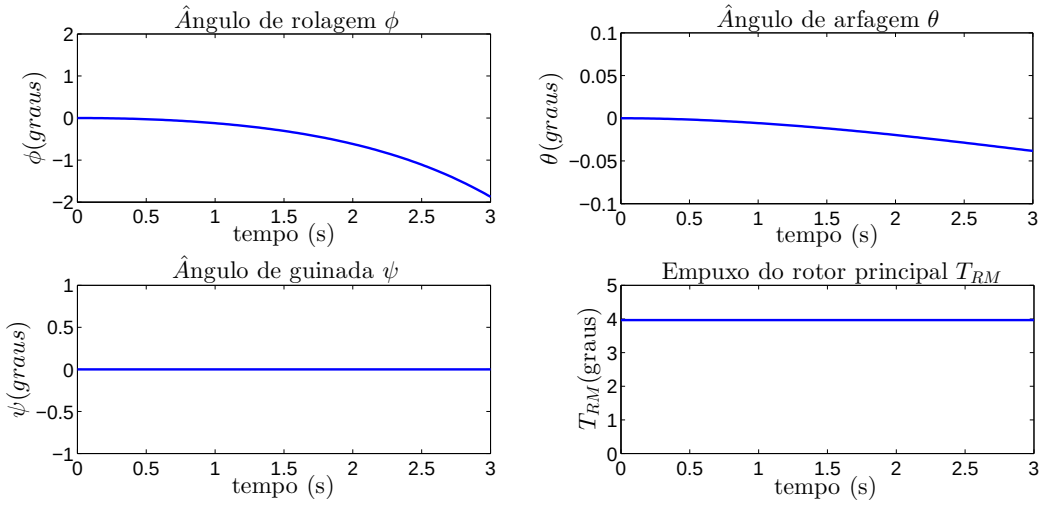


Figura 4.7: Gráficos dos ângulos de atitude do helimodelo e do empuxo do rotor principal. $U_{col} = 0.018315$

4.5.3 Influência do comando pedal

O terceiro caso de teste foi o caso em que um comando pedal não nulo,$U_{ped} = 0.01$ é aplicado ao helicóptero em vôo pairado. Utilizou-se novamente a condição inicial de vôo pairado com comando coletivo $U_{col} = 0.0018135$. Para facilitar a análise, buscou-se minimizar o movimento de rolagem observado na figura 4.7, pela aplicação de um sinal $U_{lat} = 0.0075$ que induz um torque que se opõe ao movimento de rolagem sem mecanismo de controle. O esperado para esta simulação é que os ângulos de arfagem e rolagem variem menos do que no resultado da simulação de vôo pairado, e um movimento de guinada seja induzido pelo comando U_{ped}. A figura 4.8 apresenta os resultados da simulação, em que observa-se um movimento negativo de guinada induzido pelo comando pedal, a diferença de sinal entre a entrada e saída não indica nenhuma incoerência no modelo, apenas que foram adotadas convenções de sinais diferentes entre o comando e o acionamento. Todos os movimentos observados em simulação correspondem aos esperados.

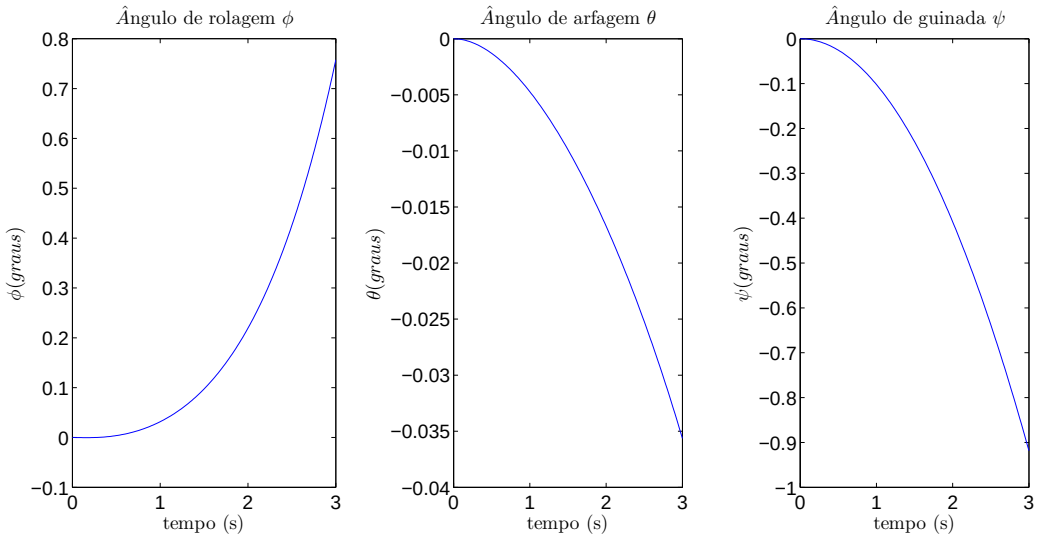


Figura 4.8: Gráficos dos ângulos de atitude do helimodelo em resposta aos comandos $U_{col} = 0.018315$, $U_{ped} = 0.01$, $U_{lat} = 0.0075$, $U_{lon} = 0$

4.5.4 Influência dos comandos cíclicos

Os últimos casos de testes foram os casos em que no instante inicial o helicóptero está na horizontal com o torque devido à gravidade anulado pelo torque do motor, e o comando cíclico é não nulo. Se o comando cíclico lateral for diferente de zero, espera-se observar um movimento de rolagem referente ao comando U_{lat}, e se o comando cíclico longitudinal for diferente de zero, espera-se observar um movimento de arfagem referente ao comando U_{lon}. O movimento de arfagem ou rolagem altera a direção do empuxo gerado pelo rotor principal, que deixa de gerar apenas torque de arfagem e passa a gerar torques de rolagem ou guinada, e não mais anula completamente o torque gerado pela gravidade. O esperado é que o comando U_{lat} provoque um movimento maior de rolagem e um movimento menor de arfagem, o comando U_{lon} provoque um movimento de arfagem maior e um movimento menor de rolagem e todos os ângulos aumentem exponencialmente, pois o próprio movimento induz mais movimento. Devido a ação do controlador de guinada, e pelo comando pedal ser nulo, espera-se que o ângulo de guinada fique aproximadamente constante.

A figura 4.9 apresenta os resultados obtidos para simulaçao do modelo com $U_{lat} = 0.1$ e $U_{lon} = 0$, em que observa-se um movimento maior de rolagem e um movimento menor de arfagem conforme esperados, enquanto o ângulo de guinada permanece próximo de zero. E a figura 4.10 apresenta o resultado obtido para a simulação com $U_{lon} = 0.1$ e $U_{lat} = 0$, em que observa-se um movimento maior de arfagem e um movimento menor de rolagem, enquanto o ângulo de guinada permanece próximo de zero. Os dois resultados correspondem ao esperado para a influência do comando cíclico.

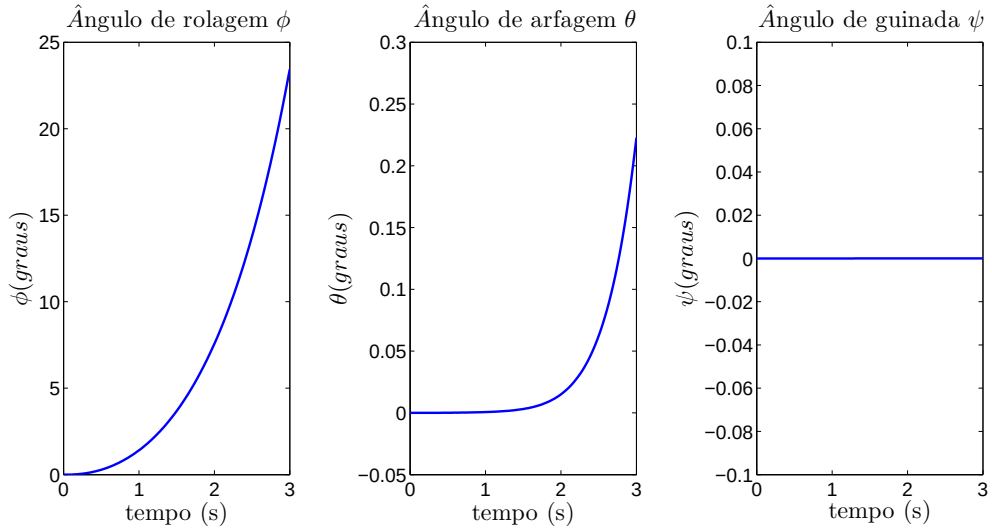


Figura 4.9: Simulação do efeito do comando coletivo lateral. $U_{col} = 0.018315$, $U_{lat} = 0.1$

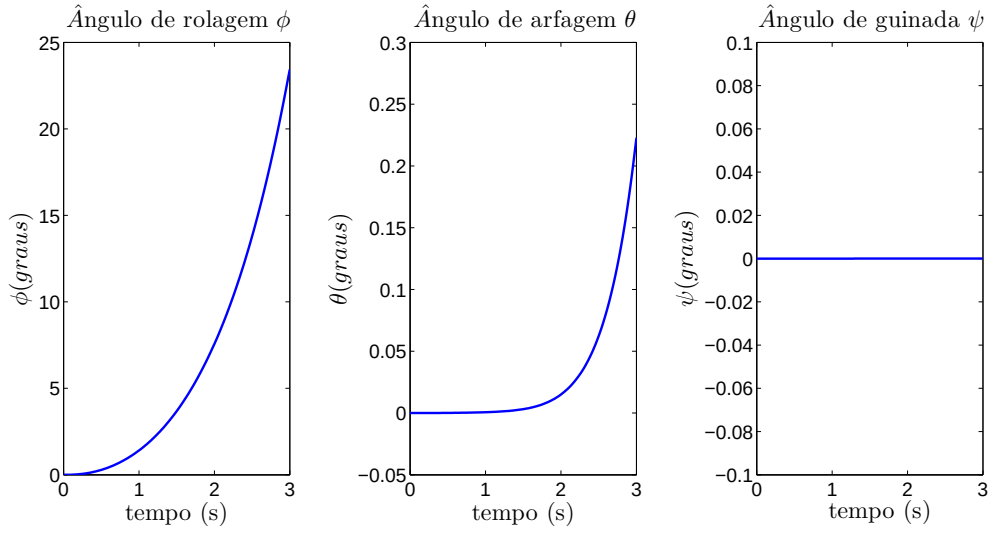


Figura 4.10: Simulação do efeito do comando coletivo longitudinal. $U_{col} = 0.018315$, $U_{lon} = 0.1$

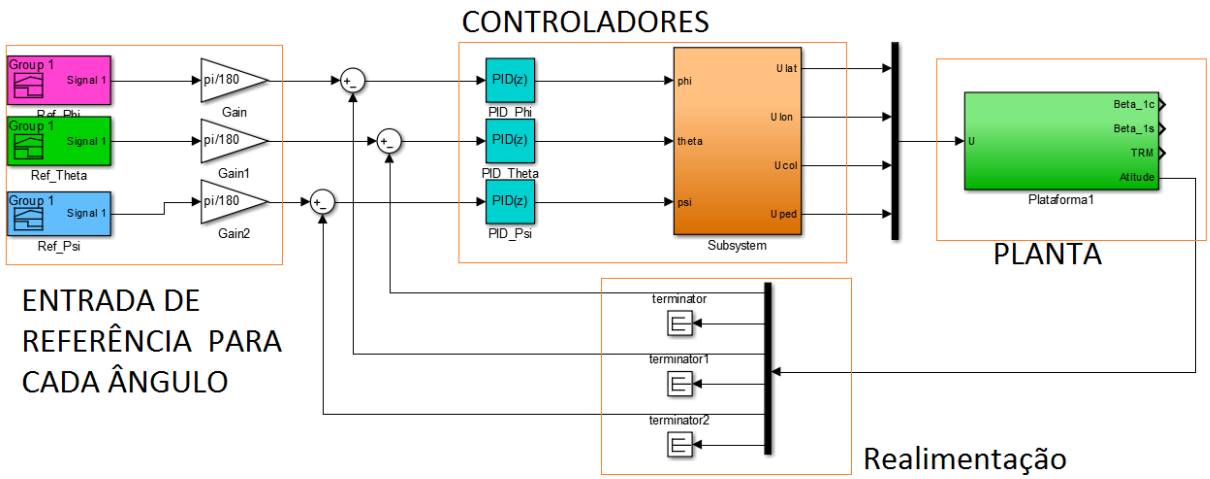


Figura 4.11: Diagrama de blocos de simulação do sistema em malha fecada

4.6 SIMULAÇÃO DO SISTEMA EM MALHA FECHADA

Após a simulação do modelo matemático obtido e validação de sua coerência, o modelo foi utilizado como planta de um sistema de controle em malha fechada, com controladores PIDs digitais e os ângulos de atitude como sinais de realimentação. O objetivo da simulação é demonstrar o comportamento do sistema em uma típica aplicação, e principalmente mostrar que controladores PIDs são capazes de estabilizar o helicóptero na plataforma e fazer com que sua atitude siga sinais de referência. A implementação desses controladores na plataforma real será útil para estabilizar o helicóptero e facilitar a execução de testes para a estimação dos parâmetros específicos do modelo matemático obtido.

A figura 4.11 apresenta o diagrama de blocos em maior nível de abstração da simulação em *Simulink* para o sistema em malha fechada. O sistema possui um sinal de referência e um controlador para cada ângulo de orientação, cada controlador recebe como entrada o sinal de erro referente ao ângulo que controla. Entre os controladores PIDs e a plataforma existe um bloco de driver de saída dos controladores, neste bloco um ganho é aplicado à saída de cada controlador e cada resultado é enviado para um sinal de controle do helimodelo diferente para que cada ângulo seja controlado apenas pelo comando que exerce maior influência sobre o ângulo em malha aberta. O ângulo de rolagem ϕ é controlado pelo comando lateral U_{lat}, o ângulo de arfagem θ é controlado pelo comando cíclico longitudinal U_{lon} e o ângulo de guinada ψ é controlado pelo comando do pedal U_{ped}. O comando coletivo U_{col} em vôo livre tem a utilidade de controlar os movimentos de translação, nos movimentos fixos à plataforma a dinâmica é simplificada mantendo-o constante para que o empuxo gerado pelo rotor principal seja constante. O diagrama de blocos do driver de saída do controlador está apresentado na figura 4.12.

As constantes dos três controladores PIDs foram obtidas através de testes experimentais até que os resultados a resposta dos ângulos aos sinais de referência apresentasse um tempo de resposta aceitável e erro de estado estacionário próximo de zero. Os valores obtidos para as constantes estão apresentados na tabela 4.6. A simulação foi feita para o caso em que primeiramente os sinais de referência estabilizam o

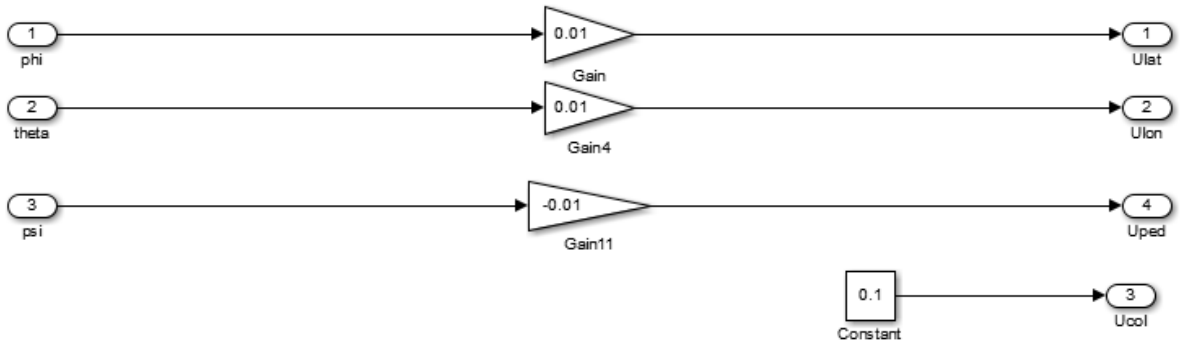


Figura 4.12: Diagrama de blocos do driver de saída dos controladores

Tabela 4.6: Constantes dos controladores PIDs

	K_p	K_i	K_d
Controlador de arfagem	30	0.01	10
Controlador de rolagem	50	40	10
Controlador de guinada	20	10	50

helimodelo em 'vôo' pairado, em seguida os sinais de referência são alterados em instantes diferentes e ao final os sinais de referência voltam para zero em instantes diferentes. O objetivo do caso de simulação é observar se os controladores são capazes de estabilizar o helimodelo em 'vôo' e permitir que as entradas controlem o estado do sistema. Os sinais de entrada utilizados estão apresentados na Figura 4.13.

A verificação do desempenho do sistema é feita observando a relação entre a entrada e a saída. Entretanto, pode ocorrer que para controlar o helimodelo, os controladores em simulação coloquem o helimodelo em um estado de operação que não é fisicamente realizável ou coerente. Por esse motivo, foram observados também os sinais de controle gerados pelos controladores e a resposta do rotor principal.

Os ângulos de orientação obtidos como resposta aos sinais de referência estão representados nos gráficos da figura 4.14. Em que observa-se que os ângulos de orientação seguem os sinais de referência, e que os efeitos do acoplamento entre a rotações é corrigido pelo controlador, o que pode ser observado no instante $t = 10s$ onde uma mudança do ângulo de guinada causa uma alteração do ângulo de rolagem, pois a atuação do movimento de guinada é feita pelo rotor de cauda que também induz um movimento de rolagem. Sem a ação dos controladores os efeitos deste movimento seriam permanentes e instabilizariam o sistema, no sistema em malha fechada o controlador de guinada corrige rapidamente essa pertubação, levando o helimodelo ao estado desejado.

A figura 4.16 apresenta os sinais de comando gerados pelos controladores e a figura 4.15 apresenta a força gerada pelo rotor principal e os ângulos de batimento do rotor principal. Observando as figuras mencionadas e os instantes em que ocorrem as variações do sinal de referência (figura 4.13), observa-se que uma mudança do sinal de referência de rolagem, altera o comando U_{lat} que provoca uma mudança no ângulo de batimento β_{1s} para induzir o movimento indicado pelo sinal de referência. Da mesma maneira,

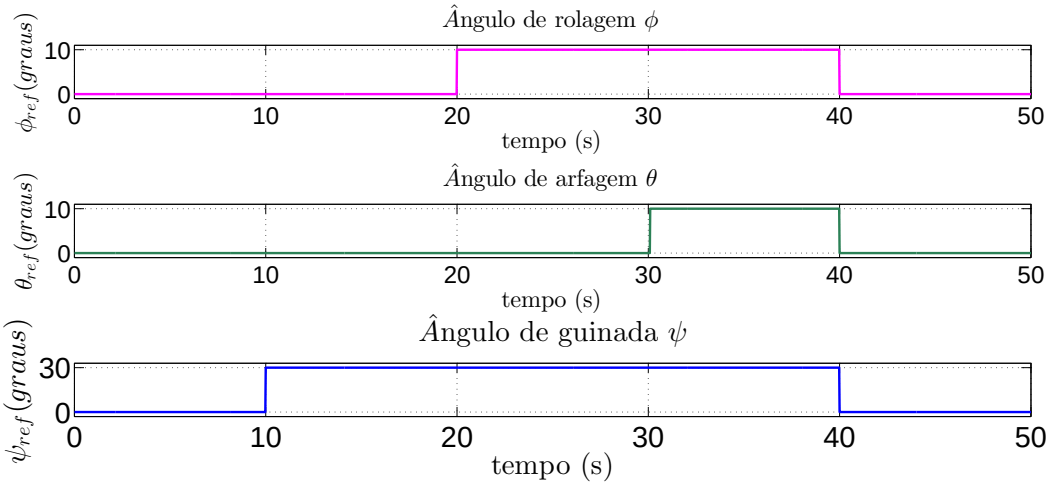


Figura 4.13: Sinais de referência da atitude

uma mudança no sinal de referência de arfagem altera o comando de U_{lon} que provoca uma alteração do ângulo de batimento β_{1c} que induz um movimento de arfagem conforme o sinal de referência. E alterações do sinal de referência de guinada provoca uma alteração no sinal altera o comando U_{ped} que interfere na força do rotor de cauda para que ocorra o movimento de guinada desejado. Nota-se também que, conforme esperado, nos instantes em que não ocorre variação dos sinais de entrada e saída, os sinais de comando do helimodelo, o empuxo e os ângulos de batimento do rotor principal permanecem constantes. O efeito da mudança simultânea de mais de um sinal de referência pode ser analisado no instante $t = 40s$, onde todos os sinais de referência são alterados e mesmo assim os ângulos de orientação seguem os sinais de referências.

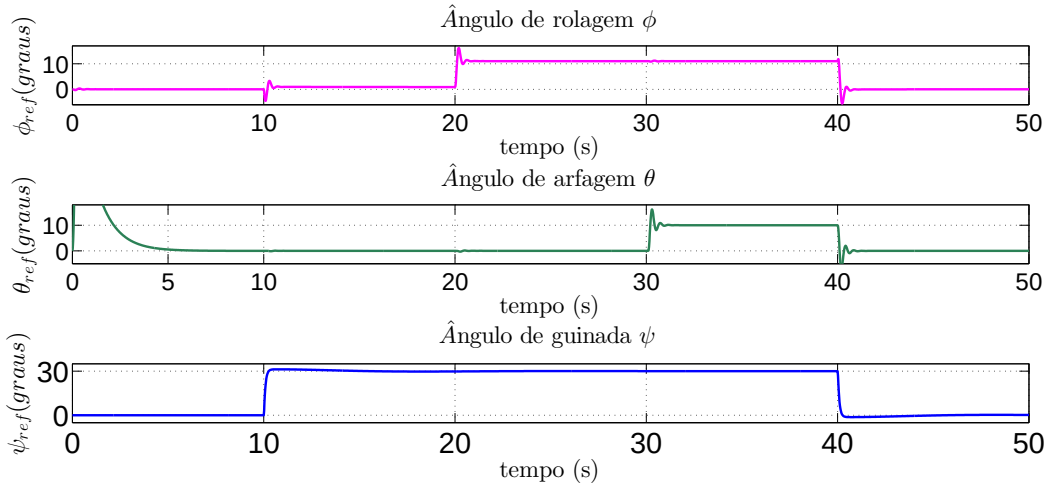


Figura 4.14: Resposta do sistema controlado aos sinais de referência

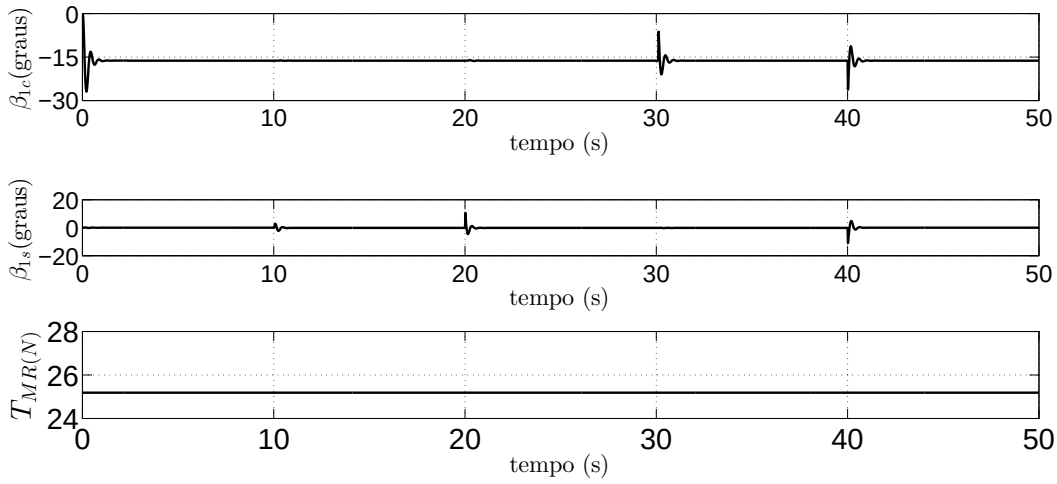


Figura 4.15: Comportamento dos atuadores

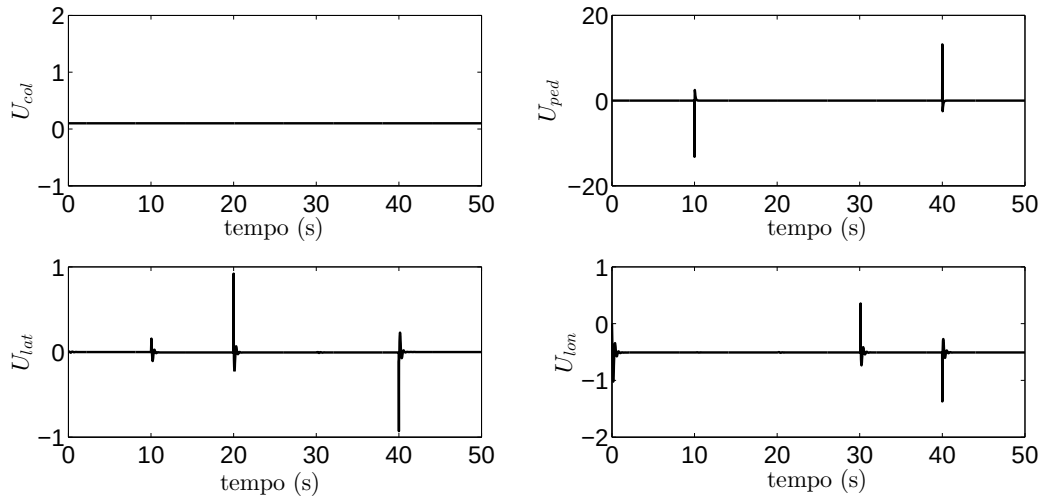


Figura 4.16: Sinais de comando gerados pelos controladores

5 CONCLUSÕES

Este trabalho apresentou as tarefas realizadas para o desenvolvimento da plataforma de teste de sistemas de controle de um helimodelo, os resultados obtidos e os testes de verificação dos resultados. Apresentou-se uma revisão bibliográfica sobre a dinâmica de helicópteros, sistemas e métodos de estimação da atitude e fusão sensorial por filtro de Kalman. Em seguida, apresentou-se o projeto da estrutura física da plataforma, o projeto da interface de aquisição de dados e atuação e o projeto do sistema de navegação para estimativa da atitude. Os resultados obtidos foram avaliados em testes e simulações, apresentou-se os testes de desempenho da interface e testes de coerência das estimativas de atitude obtidas, e simulações do modelo matemático obtido em malha aberta para verificar a coerência do modelo, e simulações em malha fechada com um controlador digital PID para cada ângulo de atitude para demonstrar um caso de utilização da plataforma e prever um método de estabilização do helimodelo para estimação dos parâmetros do modelo matemático.

A estrutura física é formada por uma base de fixação com um rolamento no topo, a base é conectada a uma junta por um eixo de transferência, o conjunto rolamento e junta permite ao helimodelo os movimentos de rotação. Inicialmente a plataforma foi construída com um modelo comercial de junta Cardan de aço, durante os testes dos sensores observou-se que essa junta interfere nas medidas do magnetômetro, degradando as estimativas de atitude. Como solução uma junta foi projetada para a plataforma, o modelo de junta projetada é formada por dois garfos conectados por cruz e rolamentos ABEC7, A cruz possui um espaço interno para encaixe da placa *SEN-10724* em seu centro, posição que corresponde ao centro de rotação da plataforma onde movimentos de translação são nulos, condição ideal para instalação do acelerômetro para medidas do vetor aceleração gravitacional. Os dois garfos possuem encaixes para potenciômetros que podem servir como fonte de medida dos ângulos de rotação. Até o momento de escrita deste relatório, a junta desenvolvida ainda estava em fabricação.

O modelo matemático obtido representa a dinâmica do helimodelo fixo à plataforma por meio da resposta dos ângulos de orientação aos sinais de controle coletivo, pedal e cíclico. Diferente dos modelos matemáticos para helicópteros convencionais, o modelo levantado considera os efeitos da barra estabilizadora adicionada a helimodelos, considera também as restrições e torques de atrito introduzidos pela plataforma que impede os movimentos de translação. O modelo matemático obtido foi publicado em [30]. Parâmetros práticos precisam ser estimados para adequação do modelo ao sistema real. O comportamento instável do helimodelo inviabiliza a estimação desses parâmetros sem a ação de um controlador, o que motivou a simulação do sistema em malha fechada com controladores digitais PIDs, pois estes controladores possuem poucos parâmetros para serem ajustados, e o ajuste pode ser feito experimentalmente. A estratégia utilizada na construção da malha de controle simulada foi separar a dinâmicas do helicóptero, fazendo que cada controlador faça o controle de um ângulo de orientação através de um sinal de comando. Como existem três ângulos de orientação e quatro sinais de comandos, optou-se por fazer o sinal de controle coletivo constante para obter um empuxo do rotor principal constante, tornando o sistema mais fácil de ser controlado. Os resultados de simulação do modelo matemático em malha fechada e em malha aberta mostram-se coerentes com o esperado e a malha de controle com três controladores PIDs mostrou-se capaz

de realizar o controle do helimodelo com bom desempenho.

A interface de aquisição e atuação, responsável por ler os dados dos sensores e enviar para o computador e receber do computador os sinais de comando e acionar os motors conforme os comandos obtidos, foi desenvolvida em dois modelos. O primeiro modelo baseado em Arduino foi a primeira versão desenvolvida devido as facilidades de desenvolvimento com o *Arduino* e a necessidade de uma versão da interface para iniciar o desenvolvimento do sistema de navegação inercial. Em seguida uma versão foi desenvolvida *Raspbery Pi*, pois dificuldades encontradas na sincronização das tarefas com o *Arduino* fizeram parecer que não era possível desenvolver a interface em *Arduino*. Durante o desenvolvimento da versão em Raspberry Pi, o problema de sincronização do Arduino foi solucionado e as duas versões de interface foram apresentadas. As interfaces foram testadas para certificar-se de que são capazes de operar com taxa de amostragem de 50Hz, e para caracterização dos resultados foram estimados os atrasos na leitura dos sensores e no acionamento dos motores. As duas versões mostraram-se capazes de operar em 50Hz, essa taxa foi escolhida como requisito pos é a taxa máxima comandos recebidos sem perda pelos servo-motores do helimodelo.

A placa de sensoriamento utilizada foi *SEN-10724* fabricada pela *Spark-fun*, que possui um acelerômetro, um magnetômetro e giroscópio, todos triaxiais. O giroscópio é utilizado na propagação temporal da atitude estimada. O acelerômetro é utilizado para medir a aceleração gravitacional e o magnetômetro é utilizado para medir o campo magnético, os vetores medidos por esses dois sensores são utilizados para estimação da atitude pelo método TRIAD. O método TRIAD estima a atitude através da comparação de vetores no sistema de coordenadas de referência e em um sistema de coordenadas que acompanha o corpo alvo. A estimativa feita com o giroscópio apresenta um erro crescente no tempo, e a estimativa feita pelo algoritmo TRIAD apresenta ruído elevado e grandes erros para movimentos bruscos. Para obter uma estiva útil, as duas estimativas são combinadas por um Filtro de Kalman Estendido Correlato(FKEC). O FKEC é a formulação do filtro de Kalman aplicável a processos não lineares que considera que o ruído das etapas de predição e correção não são necessariamente independentes.

As estimativas da atitude fornecidas pelo sistema de navegação foram testadas em um procedimento realizado manualmente sem muita precisão dos movimentos realizados, foi avaliada a coerência da magnitude e direção dos movimentos estimados em relação aos movimentos realizados no experimento. Mostrou-se também como resultado do teste os benefícios da utilização do FKEC comparando as estimativas filtradas com as estimativas sem filtragem. As estimativas obtidas mostraram-se coerentes, e foi possível a partir da observação das estimativas filtradas e não filtradas observar o efeito positivo do FKEC.

Como trabalhos futuros de aprimoramento da plataforma propõe-se a integração dos potenciômetros para auxilar no processo de estimação da atitude e servir de referência para calibração. Desenvolver uma interface com usuário para facilitar o uso da plataforma. Propõe-se também trabalhos de utilização da plataforma no desenvolvimento de helimodelos não tripulados para o teste dos controladores de atitude.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] MOREIRA, M. A. G. *Localização, modelagem e controle de um helimodelo em ambientes internos*. Dissertação (Masters thesis) — Universidade Federal de Minas Gerais, 2010.

[2] TITTERTON, D. H.; WESTON, J. L. *Strapdown inertial navigation technology*. 2nd. ed. [S.l.]: The Institute of Electrical Engineers,United Kingdom, 2004.

[3] LLC, C. R. Understanding euler angles. *http://www.chrobotics.com/library/understanding-euler-angles*.

[4] VITZILAIOS, N. I.; TSOURVELOUDIS, N. C. Altitude control of small helicopter using a prototype test bed. In: *CD Proceedings of the 5th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2008), Madeira, Portugal*. [S.l.: s.n.], 2008.

[5] ADIGBLI, P. et al. Nonlinear attitude and position control of a micro quadrotor using sliding mode and backstepping techniques. In: *3rd US-European Competition and Workshop on Micro Air Vehicle Systems (MAV07) & European Micro Air Vehicle Conference and Flight Competition (EMAV2007)*. Toulouse, France: [s.n.], 2007.

[6] BECKER, M.; BOUABDALLAH, S.; SIEGWART, R. Desenvolvimento de um controlador de desvio de obstáculos para um mini-helicóptero quadri-rotor autônomo - 1^afase: Simulação. In: . [S.l.]: CBA - Congresso Brasileiro de Automática (CBA 2006), 2006. v. 1, p. 1201–1206.

[7] WEILENMANN, M. F. *Robuste Mehrgrössen-Regelung eines Helikopters*. Tese (Phd Thesis) — Instituto de Tecnologia Federal Suísso, Zurique, 1994., 1994.

[8] MARTINS, A. S. *Instrumentação e Controle de Atitude para Helimodelo Montado em uma Plataforma de Testes*. Dissertação (Masters Thesis) — Departamento de Engenharia Elétrica, Universidade de Brasília, 2008.

[9] LIDSTONE, C. *The Gimballed Helicopter Testbed: Design, Build and Validation*. Dissertação (Masters Thesis) — Department of Electrical and Computer Engineering, University of Toronto, 2003.

[10] TANAKA, K.; OHTAKE, H.; WANG, H. O. A practical design approach to stabilization of a 3-dof rc helicopter. *IEEE Transactions on Control Systems Technology*, v. 12, n. 2, p. 315–325, 2004.

[11] Bó, A. P. L.; MIRANDA, H. H. F. *Concepção de uma plataforma experimental para estudo de controle de um modelo reduzido de helicoptero.* [S.l.], 2004.

[12] BO, A. P. L.; MIRANDA, H. H. F. *Modelamento, simulação e controle de veículos autônomos aéreos e submarinos.* [S.l.], 2005.

[13] SPANOUDAKIS, P. et al. The market for vtol uavs. *Unmanned Systems Magazine*, Sept/Oct, p. 14–18, 2003.

[14] SANTOS, W. V. dos. *Modelagem, Identificação e Controle de Altitude de um Helicóptero em Escala Reduzida*. Dissertação (Masters Thesis) — Universidade Federal do Rio de Janeiro, 2005.

[15] TAKAHASHI, N. S. *Metodologia de Desenvolvimento de um Determinador de Atitude Portátil de Baixo Custo para Interfaces Homem-Máquina*. Dissertação (Mestrado) — Universidade Estadual de Londrina, Centro de Tecnologia e Urbanismo, 2011.

[16] BRINDEIRO, G. A. *Sistema de localização por GPS/UMI para robôs móveis em abientes externos*. [S.l.], 2011.

[17] DURãO, C. R. C. *Utilização de unidades de meddias inerciais baseadas em sistemas microeletro-mecânicos em navegadores integrados*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2009.

[18] HERRERA-MAY, A. L. et al. *Resonant Magnetic Field Sensors Based On MEMS Technology*. [S.l.], 2009.

[19] GREWAL, M. S.; WEIL, L. R.; ANDREWS, A. P. *Global position systems, inertial navigation and integration*. 2nd. ed. [S.l.]: John Wiley & Sons, Inc., 2007.

[20] KUIPERS, J. B. Quaternions and rotation sequences: A primer with applications to orbits, aerospace and virtual reality. September 1999.

[21] LEFFERTS, F. L. M. E. J.; SHUSTER, M. D. Kalman filtering for spacecraft attitude estimation. *AIAA 20th Aerospace Sciences Meeting*, 1982.

[22] BLACK, H. D. A passive system for determining the attitude of satellite. *AIAA Journal*, v. 2, n. 7, 1982.

[23] SMITH, A. S. *Microeletrônica*. 4ł. ed. [S.l.: s.n.], 2004.

[24] OGATA, K. *Engenharia de Controle Moderno*. 5ł. ed. [S.l.: s.n.], 2011.

[25] BARTZ, P. *Building an AHRS using the SparkFun "9DOF Razor IMU"or "9DOF Sensor Stick"*. [S.l.], 2013.

[26] FINNE, S. I2c library. *https://projects.drogon.net/raspberry-pi/wiringpi/i2c-library/*.

[27] HENDERSON, G. Serial library. *https://projects.drogon.net/raspberry-pi/wiringpi/serial-library/*.

[28] FINNE, S. Pi-servo. *https://projects.drogon.net/raspberry-pi/wiringpi/i2c-library/*.

[29] Bó, A. P. L. *Desenvolvimento de um sistema de localização 3D para aplicaçãoes e robôs aéreos*. Dissertação (Mestrado) — Universidade de Brasília, 2007.

[30] JúNIOR, Z. R. M.; LOPES, R. V. Modelagem de um helicÓptero elÉtrico em uma plataforma 3dof. *XIX Congresso de Estudantes de Engenharia Mecânica*, Agosto 2012.

[31] METTLER, B. *Identification Modeling and Characteristics of Miniature Rotorcraft*. [S.l.]: Kluwer Academic Publishers, Boston, MA., 2003.

[32] HALD, U. B. et al. *Autonomous Helicopter - Modelling and Control*. [S.l.], 2005.

[33] KOO, T. J.; MA, Y.; SASTRY, S. S. Nonlinear control of a helicopter based unmanned aerial vehicle model. 2001.

[34] API, X. Timer management services. *http://www.xenomai.org/documentation/trunk/html/api/group__native__timer.html*.

ANEXOS

I. CÓDIGO DE PROGRAMAÇÃO DO ARDUINO OBTIDO NO DESENVOLVIMENTO DA INTERFACE

```
1  #include <Wire.h>
2  #include <Servo.h>
3  #include <avr/sleep.h>
4  #include <avr/power.h>
5  #include <stdarg.h>
6
7  //ENDEREÇOS E PARÂMETROS
8
9  //Acelerômetro
10 #define ADXL345_ADDRESS (0xA6 >>1)
11 #define ADXL345_REGISTER_XLSB (0x32)
12 #define ADXL_REGISTER_PWRCTL (0x2D)
13 #define ADXL_PWRCTL_MEASURE (1<<3)
14
15 //Giroscópio
16 #define ITG3200_ADDRESS (0xD0 >> 1)
17 #define ITG3200_REGISTER_XMSB (0x1D)
18 #define ITG3200_REGISTER_DLPF_FS (0x16)
19 #define ITG3200_FULLSCALE (0x03 << 3)
20 #define ITG3200_42HZ (0x03)
21
22 //Magnetômetro
23 #define HMC5843_ADDRESS (0x3C >> 1)
24 #define HMC5843_REGISTER_XMSB (0x03)
25 #define HMC5843_REGISTER_MEASMODE (0x02)
26 #define HMC5843_MEASMODE_CONT (0x00)
27
28 //Baud-Rate
29 #define BR 115200
30 //Número de Bytes recebidos
31 #define RXLEN  20
32 //Pino do servo de inclinação das pás do rotor principal
33 #define SERVO1 5
34 //Pino do servo de alteração do ângulo de batimento lateral
35 #define SERVO2 6
36 //Pino do servo de alteração do ângulo de batimento longitudinal
37 #define SERVO3 7
38 //Pino do servo de inclinação das pás do rotor de cauda
39 #define SERVO4 8
40 //Pino do controlador de velocidade do motor DC
41 #define MotorDC 3
42
43
44 //TDMA = 16000000/(64*Fs)-1 //Período de interrupção do Timer.
45 #define TDMA 2499
46
47 int i,j;
48 //Variáveis de armazenamento das leituras
49 int Ax=0,Ay=0,Az=0,Gx=0,Gy=0,Gz=0,Mx=0,My=0,Mz=0;
50
51 int estado=1;
52 int recebido[RXLEN];
```

```
53 int sinalmotor[5];
54 Servo servo1, servo2, servo3,servo4, DCmotor;
55
56 //Função de envio de dados para computador
57 void sprint(char *fmt, ... ){
58   char tmp[128]; // resulting string limited to 128 chars
59   va_list args;
60   va_start (args, fmt );
61   vsnprintf(tmp, 128, fmt, args);
62   va_end (args);
63   Serial.print(tmp);
64 }
65
66
67 void setup(){
68   Wire.begin();                  //inicia comunicação I2C
69   Serial.begin(BR);              //configuração do baudrate
70   delay(5);
71
72   //Configuração doaADXL345
73   Wire.beginTransmission(ADXL345_ADDRESS);
74   Wire.write(ADXL_REGISTER_PWRCTL);
75   Wire.write(ADXL_PWRCTL_MEASURE);
76   Wire.endTransmission();
77
78   //Configuração do ITG3200
79   Wire.beginTransmission(ITG3200_ADDRESS);
80   Wire.write(ITG3200_REGISTER_DLPF_FS);
81   Wire.write(ITG3200_FULLSCALE | ITG3200_42HZ);
82   Wire.endTransmission();
83
84   //Configuração do HMC5843
85   Wire.beginTransmission(HMC5843_ADDRESS);
86   Wire.write(HMC5843_REGISTER_MEASMODE);
87   Wire.write(HMC5843_MEASMODE_CONT);
88   Wire.endTransmission();
89
90   servo1.attach(MOTORDC);
91   servo2.attach(SERVO1);
92   servo3.attach(SERVO2);
93   servo4.attach(SERVO3);
94   servo5.attach(SERVO4);
95
96   cli();     //Desabilita interrupções
97
98   //CONFIGURAÇÃO DO TIMER4
99
100  TCCR4A = 0;   // reseta configuração
101  TCCR4B = 0;   // reseta configuração
102  TCNT4  = 0;   // inicia contador em 0
103
104  // Configura registrador de comparação para o período de interrupção desejado
105  OCR4A = TDMA;// = (16*10^6) / (1*1024) - 1 (must be <65536)
106
107  // Configura interrupção por comparação
108  TCCR4B |= (1 << WGM42);
109
110  // Configura divisor de clock 1:64
111  TCCR4B |= (1 << CS42);
112  // Configura  interrupções por tempo
```

```
113   TIMSK4 |= (1 << OCIE4A);
114   sei(); //Habilita interrupções
115 }
116
117
118 void loop(){
119   switch(estado){
120   case 1:
121
122     //PROCESSO DE AQUISIÇÃO
123
124     //Leitura do acelerômetro
125
126     Wire.beginTransmission(ADXL345_ADDRESS);
127     Wire.write(ADXL345_REGISTER_XLSB);
128     Wire.endTransmission();
129     Wire.beginTransmission(ADXL345_ADDRESS);
130
131     Wire.requestFrom(ADXL345_ADDRESS,6);
132
133     Ay=Wire.read();
134     Ay+=Wire.read()<<8;
135     Ax=Wire.read();
136     Ax+=Wire.read()<<8;
137     Az=Wire.read();
138     Az+=Wire.read()<<8;
139
140
141     Wire.endTransmission();
142
143 //    DESCOMENTE PARA MEDIR ATRASO NA LEITURA DO ACELERÔMETRO
144 //    Serial.print(TCNT4);
145 //    Serial.print("\n");
146
147     //Leitura do giroscópio
148
149     Wire.beginTransmission(ITG3200_ADDRESS);
150     Wire.write(ITG3200_REGISTER_XMSB);
151     Wire.endTransmission();
152     Wire.beginTransmission(ITG3200_ADDRESS);
153
154     Wire.requestFrom(ITG3200_ADDRESS,6);
155
156     Gy=(Wire.read())<<8;
157     Gy+=Wire.read();
158     Gx=(Wire.read())<<8;
159     Gx+=Wire.read();
160     Gz=(Wire.read())<<8;
161     Gz+=Wire.read();
162
163     Wire.endTransmission();
164
165 //    DESCOMENTE PARA MEDIR ATRASO NA LEITURA DO GIROSCÓPIO
166 //    Serial.print(TCNT4);
167 //    Serial.print("\n");
168
169
170     //Leitura do magnetômetro
171
172     Wire.beginTransmission(HMC5843_ADDRESS);
```

```
173       Wire.write(HMC5843_REGISTER_XMSB);
174       Wire.endTransmission();
175       Wire.beginTransmission( HMC5843_ADDRESS);
176
177       Wire.requestFrom(HMC5843_ADDRESS,6);
178
179       Mx=Wire.read()<<8;
180       Mx+=Wire.read();
181       My=Wire.read()<<8;
182       My+=Wire.read();
183       Mz=Wire.read()<<8;
184       Mz+=Wire.read();
185
186       Wire.endTransmission();
187
188   //     DESCOMENTE PARA MEDIR O ATRASO NA LEITURA DO MAGNETÔMETRO
189   //     Serial.print(TCNT4);
190   //     Serial.print("\n");
191
192       //Envio do pacote de dados de leitura para o computador
193       sprint("%u,%u,%u,%u,%u,%u,%u,%u,%u,%u\n",Ax,Ay,Az,Gx,Gy,Gz,Mx,My,Mz);
194
195   //     DESCOMENTE PARA MEDIR A O ATRASO NA EXECUÇÃO DO PROCESSO DE AQUISIÇÃO
196   //     Serial.print(TCNT4);
197   //     Serial.print("\n");
198
199       estado=0;
200       break;
201
202     case 2:
203
204       //PROCESSO DE ATUAÇÃO
205
206       //Ler dados dos sensores
207       while(Serial.available()<RXLEN);  //Espera se não houver dados disponíveis
208       for(i=0; i<RXLEN; i++){           //Ler se houver dados disponíveis
209         for(j=0;j<5;j++){
210           recebido[i]=Serial.read();
211         }
212         serilmotor[i]=atof(recebido);
213       }
214
215
216       DCmotor.write(sinalmotor[0]);
217
218       //Serial.print(TCNT4);
219       //Serial.print("\n");
220
221       servo1.write(sinalmotor[1]);
222
223       //Serial.print(TCNT4);
224       //Serial.print("\n");
225
226       servo2.write(sinalmotor[2]);
227
228       //Serial.print(TCNT4);
229       //Serial.print("\n");
230
231       servo3.write(sinalmotor[3]);
232
```

```
233       //TEMPO=TCNT4;
234       //sprint("%d\n",TEMPO);
235
236       servo4.write(sinalmotor[4]);
237
238       //TEMPO=TCNT4;
239       //sprint("%d\n",TEMPO);
240
241       estado=3;
242
243
244       break;
245     }
246     enterIdle();   //Coloca arduino em IDLE MODE
247   }
248
249   void enterIdle(void)
250   {
251       //PRIMEIRO HABILITA-SE O IDLE MODE
252
253      set_sleep_mode(SLEEP_MODE_IDLE);
254      sleep_enable();
255
256
257       //DESABILITA TODOS OS RECURSOS DESNECESSÁRIOS
258       //QUE  PODEM ENCER O IDLE MODE
259
260
261     power_adc_disable();
262     power_spi_disable();
263     power_timer0_disable();
264     power_timer2_disable();
265     power_twi_disable();
266
267     //ENTRA EM IDLE MODE
268     sleep_mode();
269
270     //QUANDO OCORRER UMA INTERRUPÇÃO DO TIMER
271     //A EXECUÇÃO CONTINUA DAQUI
272     sleep_disable(); //DESABILITA IDLE MODE
273
274     /*REABILITA TODOS OS RECURSOS */
275     power_all_enable();
276   }
277
278   //ROTINA DE INTERRUPÇÃO
279   ISR(TIMER4_COMPA_vect){
280   if(estado==0) estado=2;
281   if(estado==3) estado=1;
282   //Serial.print("interrupt\n");
283   }
```

II. CÓDIGO DA APLICAÇÃO DA INTERFACE DESENVOLVIDA PARA RASPBERRY PI

```
1
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <signal.h>
5  #include <unistd.h>
6  #include <sys/mman.h>
7  #include <native/task.h>
8  #include <native/timer.h>
9  #include <native/alarm.h>
10 #include <wiringPiI2C.h>
11 #include <wiringSerial.h>
12 #include <string.h>
13 #include "/home/pi/tcc/pi-servo/source/servo.h"
14
15 //ENDEREÇO I2C DO ADXL345
16 #define ADXL345_ADDRESS 0x53
17
18 //ENDEREÇO DOS REGISTRADOS DOS PRIMEIROS BYTES DAS MEDIDAS
19 //CONFORME OS EIXOS DE MEDIDAS ADOTADO NESTE TRABALHO
20
21 #define ADXL345_REGISTER_YLSB 0x32
22 #define ADXL345_REGISTER_XLSB 0x34
23 #define ADXL345_REGISTER_ZLSB 0x36
24
25 //REGISTRADOR E PARÂMETRO DE CONFIGURAÇÃO
26 //DO MODO DE MEDIDA CONTÍNUA
27
28 #define ADXL_REGISTER_PWRCTL 0x2D
29 #define ADXL_PWRCTL_MEASURE 0x08
30
31 //ENDEREÇO I2C DO ITG3200
32 #define ITG3200_ADDRESS 0x68
33
34 //ENDEREÇO DOS REGISTRADOS DOS PRIMEIROS BYTES DAS MEDIDAS
35 //CONFORME OS EIXOS DE MEDIDA ADOTADOS NESTE TRABALHO
36 #define ITG3200_REGISTER_YMSB 0x1D
37 #define ITG3200_REGISTER_XMSB 0x1F
38 #define ITG3200_REGISTER_ZMSB 0x21
39
40 //REGISTRADOR E PARÂMETROS DE CONFIGURAÇÃO
41 //DO FATOR DE ESCALA E DO FILTRO BAIXAS
42 #define ITG3200_REGISTER_DLPF_FS 0x16
43 #define ITG3200_FULLSCALE 0x18
44 #define ITG3200_42Hz 0x03F
45
46 //ENDEREÇO I2C DO HMC5843
47 #define HMC5843_ADDRESS 0x1E
48
49 //ENDEREÇO DOS REGISTRADOS DOS PRIMEIROS BYTES DE MEDIDAS
50 //CONFORME OS EIXOS DE MEDIDA ADOTADOS NESTE TRABALHO
51 #define HMC5843_REGISTER_XMSB 0x03
52 #define HMC5843_REGISTER_YMSB 0x05
```

```
53 #define HMC5843_REGISTER_ZMSB 0x07
54
55 //REGISTRADOR E PARÂMETRO DE CONFGURAÇÃO
56 //DO MODO DE MEDIDA CONTÍNUA
57 #define HMC5843_REGISTER_MEASMODE 0x02
58 #define HMC5843_MEASMODE_CONT 0x00
59
60 // Estrutura de medidas
61 struct IMU{
62         int a[3];  //vetor de medida do aclerômetro
63         int g[3];  //vetor de medida do giroscópio
64         int m[3];  //vetor de medida do magnetômetro
65         float motor[5];    //vetor de dados enviados para os motores
66 };
67
68 RT_TASK leitura,recebimento,atuacao;
69 RT_ALARM ALARME_1,ALARME_2;
70 RTIME now,previous;
71 int afd,gfd,mfd;
72 int serialfd;
73 ps_servo* motor1, * motor2, * motor3,* motor4,* motor5;
74 float sum=0.0,maxerro=0.0,media=0.0,erroa=0.0;
75
76 //Função da task leitura
77 void readimu(void *arg){
78         struct IMU *medidas=(struct IMU *)arg;
79
80         //previous=rt_timer_read();
81         //int contador=0;
82         while(1){
83                 //contador++;
84                 rt_alarm_wait(&ALARME_1); //Aguarda alarme
85
86         //Leitura  do acelerômetro
87
88                 (*medidas).a[0] = wiringPiI2CReadReg8(afd, ADXL345_REGISTER_XLSB)+
89                    (wiringPiI2CReadReg8(afd, ADXL345_REGISTER_XLSB+1)<<8);
90                 (*medidas).a[1] = wiringPiI2CReadReg8(afd, ADXL345_REGISTER_YLSB)+
91                    (wiringPiI2CReadReg8(afd, ADXL345_REGISTER_YLSB+1)<<8);
92                 (*medidas).a[2] = wiringPiI2CReadReg8(afd, ADXL345_REGISTER_ZLSB)+
93                    (wiringPiI2CReadReg8(afd, ADXL345_REGISTER_ZLSB+1)<<8);
94
95
96         //Leitura do giroscópio
97
98                 (*medidas).g[0] = wiringPiI2CReadReg8(gfd, ITG3200_REGISTER_XMSB+1)+
99                    (wiringPiI2CReadReg8(gfd, ITG3200_REGISTER_XMSB)<<8);
100                (*medidas).g[1] = wiringPiI2CReadReg8(gfd, ITG3200_REGISTER_YMSB+1)+
101                   (wiringPiI2CReadReg8(gfd, ITG3200_REGISTER_YMSB)<<8);
102                (*medidas).g[2] = wiringPiI2CReadReg8(gfd, ITG3200_REGISTER_ZMSB+1)+
103                   (wiringPiI2CReadReg8(gfd, ITG3200_REGISTER_ZMSB)<<8);
104
105                //Leitura do magnetômetro
106
107                (*medidas).m[0] = wiringPiI2CReadReg8(mfd, HMC5843_REGISTER_XMSB +1)+
108                   (wiringPiI2CReadReg8(mfd, HMC5843_REGISTER_XMSB )<<8);
109                (*medidas).m[1] = wiringPiI2CReadReg8(mfd, HMC5843_REGISTER_YMSB +1)+
110                   (wiringPiI2CReadReg8(mfd, HMC5843_REGISTER_YMSB)<<8);
111                (*medidas).m[2] = wiringPiI2CReadReg8(mfd, HMC5843_REGISTER_ZMSB+1)+
112                   (wiringPiI2CReadReg8(mfd, HMC5843_REGISTER_ZMSB)<<8);
```

```
113
114                 //Envio do pacote de dados de leituras para o computador
115
116                 serialPrintf(serialfd,"%d,%d,%d,%d,%d,%d,%d,%d,%d\n",
117                                 (*medidas).a[0],(*medidas).a[1],(*medidas).a[2],
118                                 (*medidas).g[0],(*medidas).g[1],(*medidas).g[2],
119                                 (*medidas).m[0],(*medidas).m[1],(*medidas).m[2],);
120                 }
121
122 }
123
124 //Função da task recebimento
125
126 void readserial(void *arg){
127         struct IMU *angulo=(struct IMU *)arg;
128         int n,m;
129         char str[5];
130         char com;
131
132     while(1){
133          rt_alarm_wait(&ALARME_2); //Aguarda ALARME_2
134
135
136      //Aguarda enquanto não houver dados disponíveis;
137          while(serialDataAvail(serialfd)<0);
138          //com=serialGetchar(serialfd);
139          //if(com=='a')
140        for(m=0;m<5;m++){         //Laço de leitura dos cinco comandos
141                 for(n=0;n<5;n++){    //Laço de leiutra dos cinco digitos de cada comando
142                   while(serialDataAvail(serialfd)<0);
143                         str[n]=serialGetchar(serialfd);
144                 }
145                 (*angulo).motor[m]=atof(str);     //Converte dado lido para float
146             }
147         }
148 }
149
150 //Função da task atuação
151 void atuadores(void *arg){
152
153         struct IMU *angulo=(struct IMU *)arg;
154         /* CONTROLE DOS MOTORES */
155
156 while(1){
157         rt_alarm_wait(&ALARME_1); //Aguarda alarme1
158
159     //Envia para os motores os comandos recebidos
160         motor1->pos= (*angulo).motor[0];
161         motor2->pos= (*angulo).motor[1];
162     motor3->pos= (*angulo).motor[2];
163     motor4->pos= (*angulo).motor[3];
164         motor5->pos= (*angulo).motor[4];
165         }
166 }
167
168 //Tratamento de sinal
169 void catch_signal(int sig){
170         rt_task_delete(&leitura);
171         rt_task_delete(&recebimento);
172         rt_task_delete(&atuacao);
```

```
173 }
174
175 int main(int argc, char* argv[]){
176         struct IMU *medidas;
177         medidas=malloc(sizeof(struct IMU));
178
179     //Tratamento de sinal
180         signal(SIGTERM, catch_signal);
181         signal(SIGINT, catch_signal);
182
183     //configura acelerômetro
184         afd=wiringPiI2CSetup(ADXL345_ADDRESS);
185         wiringPiI2CWriteReg8(afd, ADXL_REGISTER_PWRCTL, ADXL_PWRCTL_MEASURE);
186
187   //configura giroscópio
188   gfd=wiringPiI2CSetup(ITG3200_ADDRESS);
189   wiringPiI2CWriteReg8(gfd, ITG3200_REGISTER_DLPF_FS, ITG3200_FULLSCALE | ITG3200_42Hz);
190
191     //configura o magnetômetro
192         mfd=wiringPiI2CSetup(HMC5843_ADDRESS);
193         wiringPiI2CWriteReg8(mfd, HMC5843_REGISTER_MEASMODE, HMC5843_MEASMODE_CONT);
194
195         if((serialfd=serialOpen("/dev/ttyAMA0",115200))<0){
196                 fprintf(stderr,"Unable to open serial device: %s\n",strerror(errno));
197                 return 1;
198         }
199
200         if(wiringPiSetup()==-1){
201                 printf("Unable to start wiringPi");
202         }
203
204     //prepara a comunicação com os cinco motores
205
206         motor1=ps_init_servo(18);
207         ps_prepare();
208         ps_start_servo(motor1);
209
210         motor2=ps_init_servo(23);
211         ps_prepare();
212         ps_start_servo(motor2);
213
214         motor3=ps_init_servo(24);
215         ps_prepare();
216         ps_start_servo(motor3);
217
218         motor4=ps_init_servo(25);
219         ps_prepare();
220         ps_start_servo(motor4);
221
222         motor5=ps_init_servo(8);
223         ps_prepare();
224         ps_start_servo(motor5);
225
226     //Cria os objetos de Alarme
227         rt_alarm_create(&ALARME_1,"alarmesinc");
228         rt_alarm_start(&ALARME_1,50000000,20000000);
229         rt_alarm_create(&ALARME_2,"alarmesinc");
230         rt_alarm_start(&ALARME_2,75000000,20000000);
231
232     //Aloca memória para o sistema
```

```
233         mlockall(MCL_CURRENT|MCL_FUTURE);
234 
235     //Cria e inicia as três tasks
236         rt_task_create(&leitura,"readimu",0,99,0);
237         rt_task_start(&leitura,&readimu,medidas);
238         rt_task_create(&recebimento,"readserial",0,99,0);
239         rt_task_start(&recebimento,&readserial,medidas);
240         rt_task_create(&atuacao,"atuacao",0,99,0);
241         rt_task_start(&atuacao,&atuadores,medidas);
242 
243         pause();
244         rt_task_delete(&leitura);
245         rt_task_delete(&recebimento);
246         rt_task_delete(&atuacao);
247 }
```

III. ALGORITMO DE IMPLEMENTAÇÃO DO SNI EM MATLAB -*FKEC.M*

```
close all
clc;
%fclose(s)
clear all


% s=serial('com3','BaudRate',115200,'DataBits',8,'Terminator','LF');
% fopen(s)
figure(2)

%% Calculando Referência %%

load CalibreMagnetometer;
load rotayzx;
clear Fu; clear u;
biasM=[77.3798; 161.385; -144.782]

calibreM=[0.831816, 0.00763019, -0.0193070;
          0.00763019, 0.884438, -0.0228394;
         -0.0193070, -0.0228394, 0.992902];

biasA=[ -5.5000  -39.0000   -9.5000]';
kA = [ 0.0034 0.0036 0.0038]';

erro=1;

n=1000;


%% Configuração inicial, calculo do viés do giroscópios e baseas do algoritmo TRIAD
while( n<1000)
    n=n+1;
    L = fscanf(s,'%s')
    L= double(typecast(uint16(str2num(L)),'int16'))

    fprintf(s,'a');
    grav=[-L(1)-biasA(1); L(2)-biasA(2); L(3)-biasA(3)];

    giro=[L(4);-L(5);-L(6)];
    mag=calibreM*(L(7:9)'-biasM);


    if n==1;
        mediaA=grav;
        mediaM=mag;
        mediaG=giro;
    end
    erroA=(grav-mediaA)/n;
    erroG=(giro-mediaG)/n;
    erroM=(mag-mediaM)/n;

    erro=max(abs([erroA;erroG;erroM]));
```

```
53        mediaA=mediaA+erroA;
54        mediaM=mediaM+erroM;
55        mediaG=mediaG+erroG;
56    end
57    biasG=mediaG;
58
59    % REFERENCIAS TRIAD
60    gn = mediaA/norm(mediaA);
61    mn = mediaM/norm(mediaM);
62     In=gn+mn;
63     In=In/norm(In);
64     aux=gn-mn;
65     Jn=cross(In,aux/norm(aux));
66     Jn=Jn/norm(Jn);
67     Kn=cross(In,Jn);
68
69
70    %% FILTRO DE KALMAN %%
71    %load parado;
72    disp('start')
73
74    %CONDIÇÕES INICIAIS
75    delta=1;
76    dt=0.02;
77
78    P=eye(4)*5e-2+ones(4)*5e-3;
79    Q0=eye(4)*5e-1+ones(4)*5e-3;
80    R0=eye(5)*50+ones(5);
81    Eta_u=ones(3)*0.1+eye(3);
82    Eta_s=ones(6)*0.1+eye(6);
83    x(:,1)=[1 0 0 0]';
84    ANGLE(:,1)=[0 0 0]'
85    NMAX=10000;
86
87    %filename = 'testnew51.gif';
88
89    n=1;
90    a=axes();
91
92    while(n<NMAX)
93        n=n+1
94        %*****************************LER SENSORES*****************************
95    %       L = fscanf(s,'%s');
96    %       L = typecast(uint16(str2num(L)),'int16');
97    %       L= double(L);
98    %
99    %       %*****************************CALIBRAÇÃO*****************************
100   %
101   %       A(:,n)=[-L(1)-biasA(1); L(2)-biasA(2); L(3)-biasA(3)];
102   %       G(:,n)=[L(4)-biasG(1); -L(5)-biasG(2); -L(6)-biasG(3)]*(pi/180)/(14.375);
103   %       M(:,n)=calibreM*(L(7:9)'-biasM);
104         u=G(:,n);
105   %
106         A(:,n).*kA;
107
108         %*****************************PREDIÇÃO*****************************
109         W=[0        G(1,n)  G(2,n)   G(3,n);
110          -G(1,n)    0       -G(3,n)  G(2,n);
111          -G(2,n)    G(3,n)  0        -G(1,n);
112          -G(3,n)  -G(2,n)   G(1,n)   0];
```

```
113
114         norm_u=norm(u);
115         x(:,n)=(cos(norm_u*dt/2)*eye(4)+(sin(norm_u*dt/2)/norm_u)*W)*x(:,n-1);
116
117         dWQ=[x(4,n-1),-x(3,n-1), x(2,n-1);
118              x(3,n-1),x(4,n-1),-x(1,n-1);
119              -x(2,n-1),x(1,n-1),x(4,n-1);
120              -x(1,n-1),-x(2,n-1),-x(3,n-1)];
121
122         Fx=cos(norm_u*dt/2)*eye(4)+(sin(norm_u*dt/2)/norm_u)*W;
123
124         Fu=-(dt/(2*norm_u))*sin(norm_u*dt/2)*eye(4)*x(:,n-1)*u'...
125            +((2*dt/norm_u^2)*cos(norm_u*dt/2)-(4/norm_u^3)*sin(norm_u*dt/2))*W*x(:,n-1)*u'
126            +(sin(norm_u*dt/2)/norm_u)*dWQ;
127
128         Q=Fu*Eta_u*Fu'+Q0;
129         P=Fx*P*Fx'+Q;
130
131         %*****************************CORRRÇÃO*****************************
132
133                                 %**********TRIAD*********
134         An=A(:,n);norm(A(:,n));
135         Mn=M(:,n);norm(M(:,n));
136
137         Ib=An+Mn;
138         Ib=Ib/norm(Ib);
139         aux=An-Mn;
140         Jb=cross(Ib,aux/norm(aux));
141         Jb=Jb/norm(Jb);
142         Kb=cross(Ib,Jb);
143
144         MTH=[In,Jn,Kn]*[Ib,Jb,Kb]';
145
146         Y(1:4,n) = dcm2quat(MTH)';
147         Y(5,n)= 1-norm(Y(1:4,n));
148
149         if n==1
150             x(1:4,1)=Y(1:4,1);
151             x0(1:4,1)=Y(1:4,1);
152             continue
153         end
154
155
156        %  Cálculo Numérico da jacobiana Gs do algoritmo TRIAD em função das
157        %  medidas do acelerômetro e do magnetômetro.
158
159             An = [A(1,n) + delta; A(2,n); A(3,n)]; %inc diferencial em ax
160             An=An/norm(An);
161
162             Ib=An+Mn;
163             Ib=Ib/norm(Ib);
164             Jb=cross(Ib,An-Mn);
165             Jb=Jb/norm(Jb);
166             Kb=cross(Ib,Jb);
167
168             MTH=[In,Jn,Kn]*[Ib,Jb,Kb];
169
170             Y_ax(1:4,1)= dcm2quat(MTH)';
171             Y_ax(5) = 1-norm(Y_x(1:4));
172
```

```
173         An = [A(1,n); A(2,n)+delta; A(3,n)]; %inc diferencial em ay
174         An=An/norm(An);
175
176         Ib=An+Mn;
177         Ib=Ib/norm(Ib);
178         Jb=cross(Ib,An-Mn);
179         Jb=Jb/norm(Jb);
180         Kb=cross(Ib,Jb);
181
182         MTH=[In,Jn,Kn]*[Ib,Jb,Kb];
183
184         Y_ay(1:4,1)= dcm2quat(MTH)';
185         Y_ay(5) = 1-norm(Y_y(1:4));
186
187         An = [A(1,n); A(2,n); A(3,n)+delta]; %inc diferencial em az
188         An=An/norm(An);
189
190         Ib=An+Mn;
191         Ib=Ib/norm(Ib);
192         Jb=cross(Ib,An-Mn);
193         Jb=Jb/norm(Jb);
194         Kb=cross(Ib,Jb);
195
196         MTH=[In,Jn,Kn]*[Ib,Jb,Kb];
197         Y_az(1:4,1)= dcm2quat(MTH)';
198         Y_az(5) = 1-norm(Y_z(1:4));
199
200
201         An=A(:,n)/norm(A(:,n));
202
203         Mn = [M(1,n)+delta; M(2,n); M(3,n)]; %inc diferencial em mx
204         Mn=Mn/norm(Mn);
205
206         Ib=An+Mn;
207         Ib=Ib/norm(Ib);
208         Jb=cross(Ib,An-Mn);
209         Jb=Jb/norm(Jb);
210         Kb=cross(Ib,Jb);
211
212         MTH=[In,Jn,Kn]*[Ib,Jb,Kb];
213
214         Y_mx(1:4,1)= dcm2quat(MTH)';
215         Y_mx(5) = 1-norm(Y_x(1:4));
216
217         Mn = [M(1,n); M(2,n)+delta; M(3,n)]; %inc diferencial em my
218         Mn=Mn/norm(Mn);
219
220         Ib=An+Mn;
221         Ib=Ib/norm(Ib);
222         Jb=cross(Ib,An-Mn);
223         Jb=Jb/norm(Jb);
224         Kb=cross(Ib,Jb);
225
226         MTH=[In,Jn,Kn]*[Ib,Jb,Kb];
227
228         Y_my(1:4,1)= dcm2quat(MTH)';
229         Y_my(5) = 1-norm(Y_y(1:4));
230
231         Mn = [M(1,n); M(2,n); M(3,n)+delta]; %inc diferencial em mz
232         Mn=Mn/norm(Mn);
```

```
233
234             Ib=An+Mn;
235             Ib=Ib/norm(Ib);
236             Jb=cross(Ib,An-Mn);
237             Jb=Jb/norm(Jb);
238             Kb=cross(Ib,Jb);
239
240             MTH=[In,Jn,Kn]*[Ib,Jb,Kb];
241
242             Y_mz(1:4,1)= dcm2quat(MTH)';
243             Y_mz(5) = 1-norm(Y_z(1:4));
244
245             Gs=[(Y_ax-Y(:,n))/delta, (Y_ay-Y(:,n))/delta, .. (Y_az-Y(:,n))/delta,...
246               ...(Y_mx-Y(:,n))/delta, (Y_my-Y(:,n))/delta, (Y_mz-Y(:,n))/delta];
247
248             Hx=[eye(4); -x(1:4,n)'/norm(x(1:4,n))];
249
250             R = Gs*Eta_s*Gs'+R0; %5x5
251
252             K=(P*Hx')/(Hx*P*Hx'+R);
253
254             x(:,n)=Y(1:4,n) ;x(:,n)+K*(Y(:,n)-Hx*x(:,n));
255
256             P=(eye(4)-K*Hx)*P*(eye(4)-K*Hx)'+K*R*K';
257
258             %fprintf(s,'a %5f%5f%5f%5f',Y(:,n));
259
260
261       %%%%% PLOT
262       if mod(n,10)==0
263             vec=quatrotate(x(:,n)',eye(3));
264
265
266             v1=vec(:,1)*30+vec(:,2)*10+vec(:,3)*1;
267             v2=vec(:,1)*30-vec(:,2)*10+vec(:,3)*1;
268             v3=-vec(:,1)*30-vec(:,2)*10+vec(:,3*1);
269             v4=-vec(:,1)*30+vec(:,2)*10+vec(:,3)*1;
270
271    cubex=[ v1(1),v1(1),v1(1),-v1(1),-v1(1),-v1(1);
272             v2(1),v4(1),v2(1),-v2(1),v3(1),-v2(1);
273             v3(1),-v2(1),-v4(1),v4(1),v2(1),-v3(1);
274             v4(1),-v3(1),-v3(1),v3(1),-v4(1),-v4(1)];
275
276    cubey=[ v1(2),v1(2),v1(2),-v1(2),-v1(2),-v1(2);
277             v2(2),v4(2),v2(2),-v2(2),v3(2),-v2(2);
278             v3(2),-v2(2),-v4(2),v4(2),v2(2),-v3(2);
279             v4(2),-v3(2),-v3(2),v3(2),-v4(2),-v4(2)];
280
281    cubez=[ v1(3),v1(3),v1(3),-v1(3),-v1(3),-v1(3);
282             v2(3),v4(3),v2(3),-v2(3),v3(3),-v2(3);
283             v3(3),-v2(3),-v4(3),v4(3),v2(3),-v3(3);
284             v4(3),-v3(3),-v3(3),v3(3),-v4(3),-v4(3)];
285
286             fill3(cubex,cubey,cubez,[1 2 3 4 5 6])
287             axis([-50 50 -50 50 -50 50]);
288             drawnow
289       end
290    end
291
292    fclose(s);
```