Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Departamento de Engenharia Elétrica

Trabalho de Conclusão de Curso

Construção e Projeto de Controle de um Aero Pêndulo utilizando *Model-Based Design*

Alexsandro Ferreira de Barros Júnior

Campina Grande - PB
Dezembro de 2019

Alexsandro Ferreira de Barros Júnior

Construção e Projeto de Controle de um Aero Pêndulo utilizando *Model-Based Design*

Trabalho de Conclusão de Curso submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletricista.

Área de Concentração: Controle e Automação

Universidade Federal de Campina Grande - UFCG

Centro de Engenharia Elétrica e Informática - CEEI

Departamento de Engenharia Elétrica - DEE

Coordenação de Graduação em Engenharia Elétrica - CGEE

Rafael Bezerra Correia Lima, D.Sc.

(Orientador)

Campina Grande - PB

Dezembro de 2019

Alexsandro Ferreira de Barros Júnior

Construção e Projeto de Controle de um Aero Pêndulo utilizando *Model-Based Design*

Trabalho de Conclusão de Curso submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletricista.

Aprovado em _____ /_____ /_____

Professor Avaliador
Universidade Federal de Campina Grande
Avaliador

Rafael Bezerra Correia Lima
Universidade Federal de Campina Grande
Orientador

Campina Grande - PB
Dezembro de 2019

Este trabalho é dedicado aos meus pais, Elzeli e Alexsandro, que sempre lutaram para que eu tivesse um futuro digno e sempre me apoiaram nas decisões que tomei para o meu futuro, e a todos os meus amigos e familiares que estiveram do meu lado durante toda a jornada.

"Control,
control,
you must learn control!"
(Master Yoda)

Resumo

A realização do presente trabalho tem como objetivo escrever a aplicação do *Model-Based Design* na modelagem e controle de um Aero Pêndulo. O MATLAB©/Simulink© foi escolhido como *software* de geração de código automático para a implementação da leitura de sensores, acionamento dos atuadores e também dos controladores para o sistema em malha fechada, com isso conseguimos ajustar a tensão enviada para o motor CC por um sinal PWM de forma que o Aero Pêndulo atinja uma posição angular desejada. Os controladores utilizados, P, PD e PID, foram sintonizados utilizando uma ferramenta presente no MATLAB© que permite o ajuste da dinâmica de resposta da planta controlada. O objetivo dos métodos utilizados foi otimizar o tempo de desenvolvimento do produto final.

Palavras-chave: MATLAB©, Simulink©, *Model-Based Design*, Aero Pêndulo.

Abstract

The realization of the present work has the purpose to describe the usage of Model-Based-Design for modeling and control of a Aero Pendulum. MATLAB©/Simulink© was chosen as automatic code generation software to implement the measurement of sensors, actuators activation and also the controllers of the closed loop system, i.e., adjusting the voltage sent to the DC motor by a PWM signal enables to control the angular position of the Aero Pendulum. The P, PD and PID controllers used were tuned using a toolbox of MATLAB© that allows you to adjust the controlled pant response dynamics. The used methods objectives were to optimize the final product time development.

Keywords: MATLAB©, Simulink©, Model-Based Design, Aero Pendulum.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ABNT	Associação Brasileira de Normas Técnicas
MBD	*Model-Based Design*
HDL	*Hardware Description Language*
CC	*Corrente Contínua*
RPM	*Rotações por Minuto*
MOSFET	*Metal Oxide Semiconductor Field Effect Transistor*
PWM	*Pulse Width Modulation*
LIEC	Laboratório de Instrumentação Eletrônica e Controle
MIL	*Model-in-the-Loop*
SIL	*Software-in-the-Loop*
PIL	*Processor-in-the-Loop*
HIL	*Hardware-in-the-Loop*
SIMC	*Skogestad Internal Model Control*
PID	*Proportional Integral Derivative*
IoT	*Internet of Things*

Sumário

1 Introdução

A experiência em laboratórios é uma parte fundamental no ensino de engenharia. Com o avanço tecnológico presenciado nas últimas duas décadas se torna necessário o desenvolvimento de novas plataformas de testes que abordem os assuntos pertinentes a um curso de engenharia (ENIKOV; CAMPA, 2012). Plataformas reais e simulações apresentam pontos positivos e negativos, dito isso, ao juntar os dois métodos consegue-se obter um resultado ainda melhor durante a realização de experimentos na área de Engenharia.

O aero pêndulo foi desenvolvido para que permita uma aproximação do aluno com a planta estudada, de forma a facilitar o entendimento da implementação de um controlador, utilizando da matemática envolvida, assim como visualizando o funcionamento do sistema controlado. Com equipamentos de fácil aquisição, como o Arduino, potenciômetros e motores, foi possível produzir uma plataforma educacional que, com o acoplamento do hardware com o software MATLAB©/Simulink© permite uma implementação ágil do sistema de controle.(BARROS et al., 2019)

Na Figura 1 temos uma foto real da plataforma desenvolvida, onde o Arduino utilizado está posicionado dentro da base metálica.

Figura 1 – Fotografia da plataforma educacional desenvolvida

Fonte: Elaboração Própria

Uma das características fundamentais desse trabalho é a utilização do *Model-Based Design*, que consiste na aplicação de modelos para as simulações facilitando a etapa de prototipagem física (AARENSTRUP, 2015), para isso utilizou-se o Simulink© como uma ferramenta de simulação, que, além disso, também diminuiu o tempo de codificação, pois nele é possível gerar automaticamente o código em C para o Arduino a partir do diagrama de blocos do sistema (MATHWORKS, 2016).

Além disso, durante o período de desenvolvimento da plataforma foi encontrada uma necessidade em organizar o desenvolvimento do projeto, permitindo que o mesmo fosse executado em menos tempo e de forma mais organizada. Com isso, foi aplicado o método V-Model que possibilita a realização de testes após o fim de uma etapa de projeto e desenvolvimento, validando e verificando cada um dos processos realizados.

1.1 Objetivos

O objetivo ao realizar o trabalho foi desenvolver uma plataforma educacional utilizando o método de desenvolvimento *Model-Based Design*, a partir da modelagem dinâmica serão realizadas simulações de malha aberta e malha fechada a fim de encontrar um ponto ótimo de funcionamento de forma rápida.

A maior parte do processo de desenvolvimento será feita utilizando o MATLAB©/ Simulink©, pois nele são encontradas ferramentas que automatizam todo o processo como a simulação da planta, sintonia dos controladores e monitoramento em tempo real.

1.2 Organização do Trabalho

O trabalho está estruturado em 6 capítulos, incluindo este introdutório, conforme a seguir.

No **Capítulo 2** são introduzidas as teorias de *Model-Based Design* e *V-Model*, assim como apresenta o método de desenvolvimento com geração de código automático, onde são detalhados as vantagens e desvantagens da utilização desses métodos de desenvolvimento e explicar todo o processo da parte de pesquisa, caracterização, simulações até chegar aos resultados finais.

No **Capítulo 3** é descrita a plataforma do aero pêndulo e os componentes utilizados para a montagem da plataforma educacional proposta.

No **Capítulo 4** é feita a modelagem dinâmica da planta a ser controlada (1 grau de liberdade) e de um modelo futuro com dois graus de liberdade. Também serão mostrados resultados das simulações computacionais no MATLAB©/Simulink© mostrando o funcionamento da planta em malha aberta pelo gráfico da resposta em degrau.

No **Capítulo 5** são introduzidas as teorias de controle utilizadas no desenvolvimento da plataforma e ferramentas de sintonia presentes no MATLAB©/Simulink©. Com isso podemos então realizar as simulações de forma a analisar o funcionamento da planta em malha fechada e escolher o melhor método de controle.

No **Capítulo 6** são apresentados os resultados alcançados pelo trabalho.

No **Capítulo 7** são mostradas as conclusões do trabalho e propostas futuras visando complementar as atividades desenvolvidas.

2 Projeto Baseado em Modelo

Este capítulo é voltado para o estudo da metodologia *Model-Based Design* utilizada no desenvolvimento da plataforma e como será realizado o teste de cada etapa do desenvolvimento usando o método *V-Model*.

2.1 Introdução Teórica

No desenvolvimento de novas tecnologias é comum o uso de protótipos físicos, que são utilizados para realização de testes e validações, e documentos que descrevem o funcionamento do sistema desenvolvido.

Em alguns casos a utilização de protótipos torna o projeto muito mais caro, além disso, a interpretação dos requisitos para elaboração de códigos ou realização de outras funcionalidades, são fortemente dependentes da pessoa ou equipe que o faz, levando a um consumo de tempo maior para finalização do projeto.

A metodologia *Model-Based Design* (MBD) é uma abordagem centrada em modelos para o desenvolvimento de sistemas de controle, processamento de sinais, comunicações entre outros sistemas dinâmicos (AARENSTRUP, 2015). Ao invés de partir de protótipos físicos ou especificações textuais, o MBD foca em um modelo que inclui todos os componentes relevantes do sistema: algoritmos, lógica de controle, componentes físicos e propriedade intelectual. Depois de desenvolvido, o modelo torna-se fonte de várias saídas, incluindo relatórios, código C e HDL.

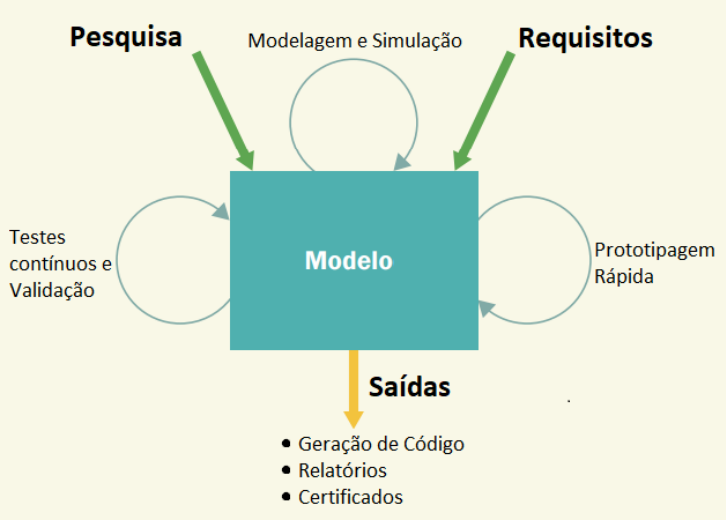
O Projeto Baseado em Modelo permite o desenvolvimento em nível de sistema ou componente, simulações, geração automática de código e testes contínuos. Um fluxo de trabalho típico está representado na Figura 2.

Com isso, torna-se possível a realização de simulações de algoritmos ou da conexão entre componentes elétricos, mecânicos e embarcados, antes mesmo do código embarcado ser escrito ou que o hardware seja fabricado e esteja disponível para testes (CHAUHAN, 2018).

2.2 Testes Estruturados

Uma das grandes vantagens no MBD é que os testes podem começar ainda na fase de requisitos, possibilitando simular as especificações e verificar se os critérios são atendidos. Como resultado, os defeitos são localizados e solucionados nas fases iniciais do projeto, reduzindo o custo total do desenvolvimento (LIN, 2011). No processo de

Figura 2 – Representação do fluxo de trabalho do *Model-Based Design*



Fonte: (AARENSTRUP, 2015).

desenvolvimento com MBD podem ser aplicados os testes em *Model-In-the-Loop* (MIL), *Software-In-the-Loop* (SIL), *Processor -In-the-Loop* (PIL) e *Hardware-In-the-Loop* (HIL).

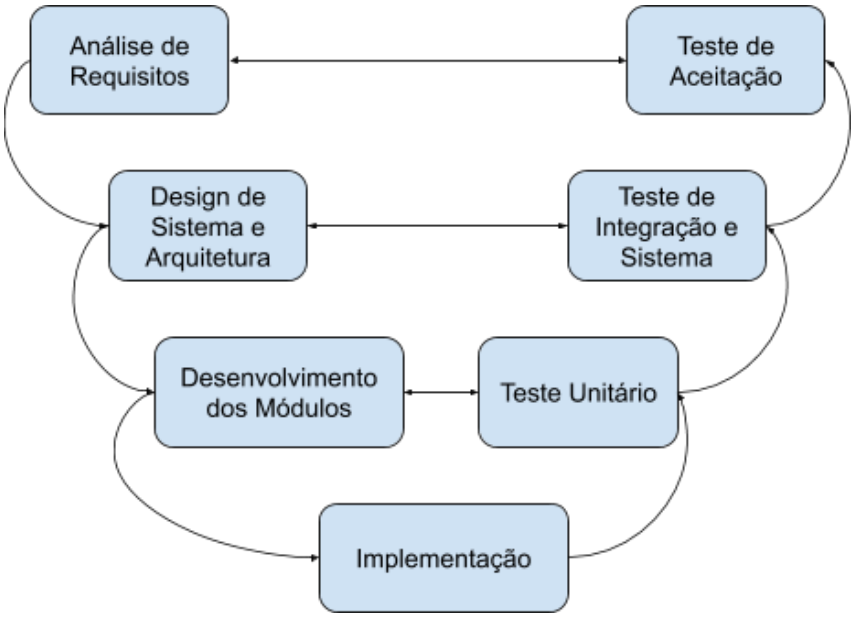
- **MIL**: consiste em realizar simulações dos modelos da planta e do controlador em um mesmo ambiente computacional. Essas simulações são realizadas em linguagem nativa da ferramenta utilizada no desenvolvimento dos modelos, permitindo assim, ter um completo domínio sobre todo o sistema simulado.
- **SIL**: as simulações dos modelos da planta e do controlador ainda são realizadas no mesmo ambiente computacional. Porém a linguagem utilizada já é a nativa do dispositivo alvo, nesse caso C/C++.
- **PIL**: o modelo do controlador, convertido em código objeto (C/C++), passa a ser executado pelo microcontrolador, enquanto a planta continua a ser simulada em linguagem nativa da ferramenta utilizada para o desenvolvimento dos modelos. Uma camada de código é adicionada ao *firmware* para permitir o envio/recebimento de dados entre a ferramenta de simulação e o microcontrolador.
- **HIL**: nesse caso um computador de tempo real executa o código da planta e fornece interfaces de entrada e saída similares ao sistema real, enquanto o algoritmo de controle é executado no microcontrolador.
- **Sistema Real**: por fim, os testes são realizados com o sistema embarcado interligado à planta, possibilitando verificar o funcionamento da lógica de controle no microcontrolador em conjunto como as interfaces de entrada/saída.

O objetivo com esses testes é realizar a verificação e validação dos requisitos do sistema em diferentes níveis do desenvolvimento, à medida que o andamento da solução avança, o funcionamento e o desempenho do sistema são comparados em cada fase de testes.

2.3 *V-Model*

O *V-Model*, também conhecido como Verificação e Validação, é uma metodologia de desenvolvimento de projetos, utilizada principalmente na área de *software*, onde a execução do processo acontece de forma sequencial.

Figura 3 – Diagrama *V-Model*



Fonte: Elaboração Própria.

As fases de desenvolvimento, lado esquerdo do diagrama, possuem uma fase de teste associada, lado direito do diagrama.

Na primeira fase do desenvolvimento, denominada Análise de Requisitos, deve-se realizar o levantamento de requisitos, onde as funções do sistema são descritas.

Em seguida o Design de Sistema e Arquitetura é realizado, onde todo o sistema é projetado especificando hardware utilizados, como o software será desenvolvido e as devidas comunicações necessárias.

Na fase de Desenvolvimento dos Módulos o funcionamento de cada parte do sistema é especificado detalhadamente, conhecido também como design de baixo nível.

Com tudo definido, é iniciada a fase de Implementação onde todo o sistema é codificado e montado.

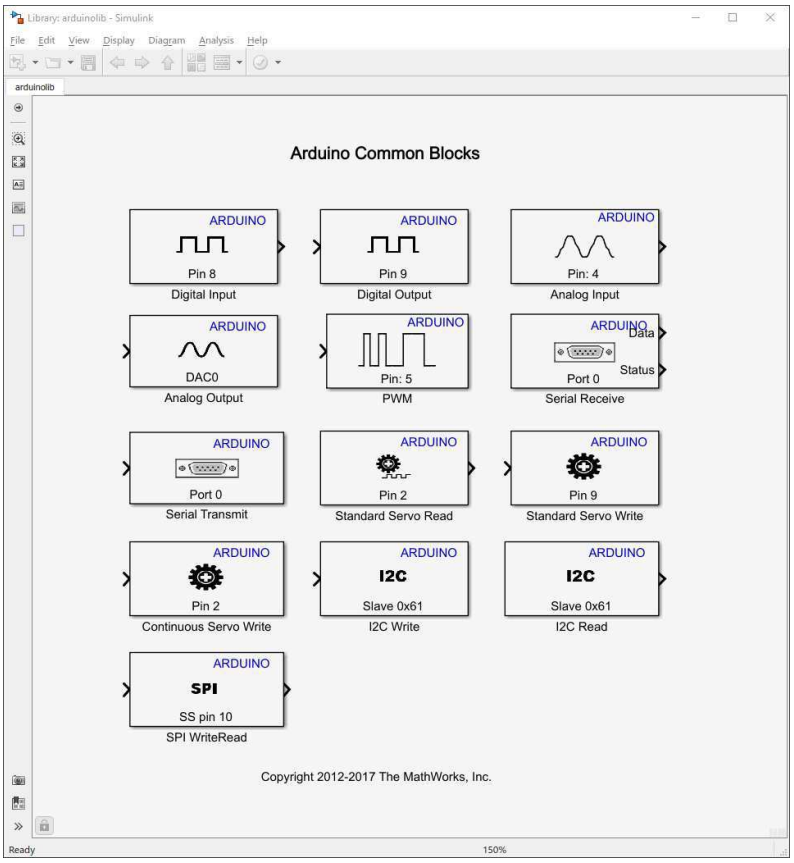
A fase de testes é dividida em Testes Unitários, onde cada módulo é testado individualmente, Testes de Integração e Sistema, que validam as comunicações entre os módulos e compatibilidade de dados, e Testes de Aceitação, onde analisa-se se o sistema atende aos requisitos definidos na fase inicial.

2.4 Geração Automática de Código

A geração automática do código executável, a partir do modelo simulado, é um dos grandes atrativos para a adoção da metodologia MBD na implementação de sistemas complexos. Um dos propósitos é a redução do tempo de desenvolvimento em relação à codificação manual bem como a mitigação de erros provenientes dela.

Na elaboração da solução apresentada neste trabalho, os modelos foram implementados no ambiente do MATLAB©/Simulink© e convertidos automaticamente em código executável para o microcontrolador Arduino Uno R3 a partir do pacote *Simulink© Support Package for Arduino© Hardware*, na Figura 4 podemos ver como é apresentado o pacote. (MATHWORKS, 2016).

Figura 4 – Representação da biblioteca do pacote de suporte para Arduino© no Simulink©



Fonte: Simulink© R2018a.

O pacote de suporte para Arduino permite realizar a programação do microcontrolador com *scripts* do MATLAB©, pois as funções encontradas na IDE do Arduino, como por exemplo *digitalWrite()*, *analogRead()*, entre outras, são adicionadas à biblioteca do MATLAB©. No Simulink© também são encontradas as mesmas funções, porém dispostas em formato de blocos para conexão com qualquer elemento nativo tornando a programação para o Arduino© bem intuitiva. Com isso, podemos utilizar o modelo desenvolvido em MBD para o sistema e acoplar os elementos do Arduino© diretamente.

O Simulink© cria uma conexão de comunicação entre o *hardware* e o computador permitindo que o código gerado seja transmitido para o Arduino© e que os dados possam fluir entre os dois, assim podemos ter o monitoramento constante do que está acontecendo com o modelo criado ao ser aplicado no sistema embarcado.

Para uma introdução à utilização do Arduino© integrado ao Simulink© utilizamos a plataforma de desenvolvimento Grove, que fornece componentes dispostos em módulos e um *shield* para Arduino. Desenvolvida pela Seeed Studio, empresa de produtos tecnológicos focada em *hardwares* para IoT ('Internet das Coisas'). A plataforma tem como objetivo facilitar a prototipagem com conexões rápidas e simplificar o processo de aprendizagem para iniciantes em programação para Arduino©. Cada módulo, que podem ser vistos na Figura 5, possui um componente diferente, *LED*, potenciômetro, sensor de toque, e outros que são conectados ao *shield* de forma direta por um padrão de comunicação desenvolvido para a plataforma (SEEEDSTUDIO, 2012).

Figura 5 – Imanges da plataforma de desenvolvimento Grove

Fonte: Seeed Studio.

3 Descrição da Planta

Este capítulo é voltado para explicar o funcionamento do aero pêndulo e mostrar os componentes utilizados para a confecção da plataforma de testes.

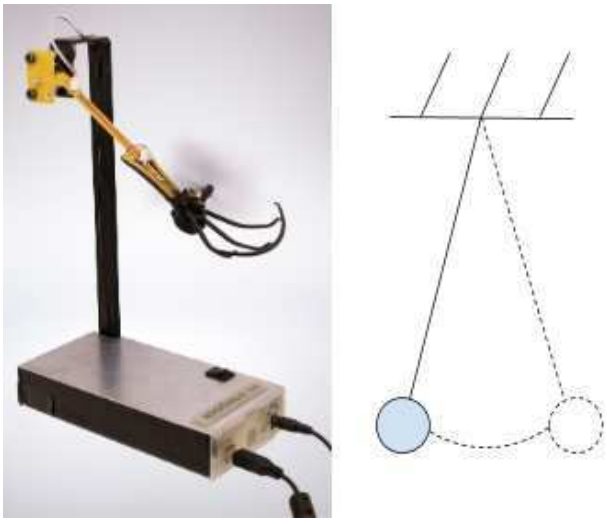
3.1 Princípios de Funcionamento

Para explicar o funcionamento do aero pêndulo pode-se utilizar como analogia um pêndulo simples, uma massa acoplada a um fio inextensível e fixo em uma extremidade a uma superfície (JOB; JOSE, 2015).

No sistema desenvolvido o atuador, formado por um conjunto motor CC e hélice, faz com que o eixo se movimente, produzindo uma força que atua de forma contrária à gravidade, dependendo da posição do motor. Tal sistema é conhecido como aero pêndulo. Aplicando um sistema de controle ao aero pêndulo, pode-se então determinar qual a intensidade da força produzida pelo motor para que o eixo assuma uma posição desejada.

É ilustrado, na Figura 6, o aero pêndulo em funcionamento ao lado de um pêndulo simples para ilustrar a semelhança.

Figura 6 – Foto do Aero Pêndulo (a) e Representação do Pêndulo Simples (b)



(a) (b)
Fonte: Elaboração Própria

Normalmente os objetivos dos sistemas de controle se resumem a rastreamento de referência ou rejeição de perturbações.

Para atingir tais objetivos, torna-se necessária a presença de um componente controlável, no caso do aero pêndulo o atuador é o conjunto de motor e hélice, alimentado por um sinal PWM. É possível controlar a velocidade do motor variando a largura do pulso.

Como explicado anteriormente, o aero pêndulo funciona de tal forma que a posição do eixo varia no percurso de uma circunferência, logo, se estamos em 0° e deseja-se mudar para 45°, a velocidade do motor é aumentada para que o eixo saia da posição estacionária e, ao alcançar o ângulo desejado deve-se manter uma velocidade constante de forma que a força da gravidade não leve o eixo à posição inicial.

3.2 Características Físicas

O aero pêndulo desenvolvido é composto por um conjunto motor hélice acoplado a um potenciômetro através de uma haste de 18cm.

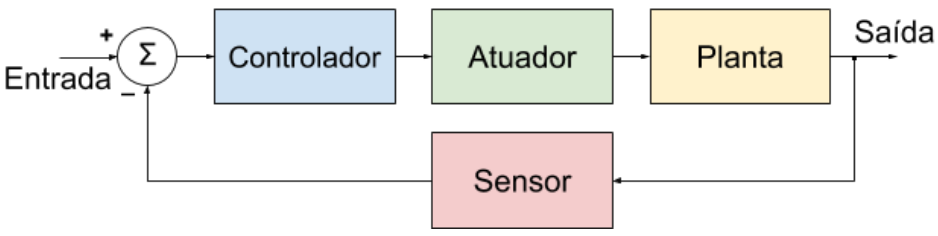
A haste que suporta o conjunto motor e hélice é constituída de uma placa de fibra de vidro, onde as trilhas presentes na mesma servem para alimentação do motor. O potenciômetro é fixado num braço de chapa de aço. Esse material também é usado na caixa presente na base da plataforma onde o Arduino© Uno e o circuito de alimentação do motor estão incluídos. Uma chave liga e desliga foi adicionada para facilitar a utilização da plataforma.

Existem limitadores de movimento angular na haste para que eventuais distúrbios não danifiquem a plataforma.

3.3 Instrumentação

O aero pêndulo desenvolvido neste trabalho apresenta componentes que são fundamentais para o funcionamento da maior parte dos sistemas de controle. Na Figura 7 é mostrado o diagrama de blocos do sistema de controle utilizado para o aero pêndulo.

Figura 7 – Diagrama do sistema de controle



Fonte: Elaboração Própria

3.3.1 Sensor

Existem diversos modelos de sensores angulares no mercado, como magnético, óptico, entre outros, normalmente utilizados em robótica e na indústria automotiva. O potenciômetro linear de 10 kΩ foi escolhido pois funciona como uma resistência variável. De acordo com o seu *datasheet* essa variação é linear em relação ao ângulo de giro do eixo, que possui uma limitação de 270 graus, com isso, conseguimos utilizá-lo para realizar a medição do ângulo de saída.

3.3.2 Atuador

O motor de corrente contínua sem escovas e sem núcleo é muito utilizado em aplicações com drone por serem compactos, consumirem pouca energia e serem facilmente substituídos devido ao preço baixo e alta disponibilidade no mercado. Na Figura 8 pode ser visto a montagem do motor com hélice usados na plataforma.

O motor caracteriza-se por possuir dimensão de 7 mm de diâmetro por 20 mm de comprimento, engrenagem de 9 dentes, fonte de alimentação de 5 V CC e corrente de 2 A. Com aproximadamente 5250 RPM na hélice, fazendo então a relação de engrenagens, onde a engrenagem da hélice possui 55 dentes, obtemos a rotação de aproximadamente 32000 RPM para o motor.

Figura 8 – Imagem do conjunto motor e hélice

Fonte: Elaboração Própria

3.3.3 Arduino©

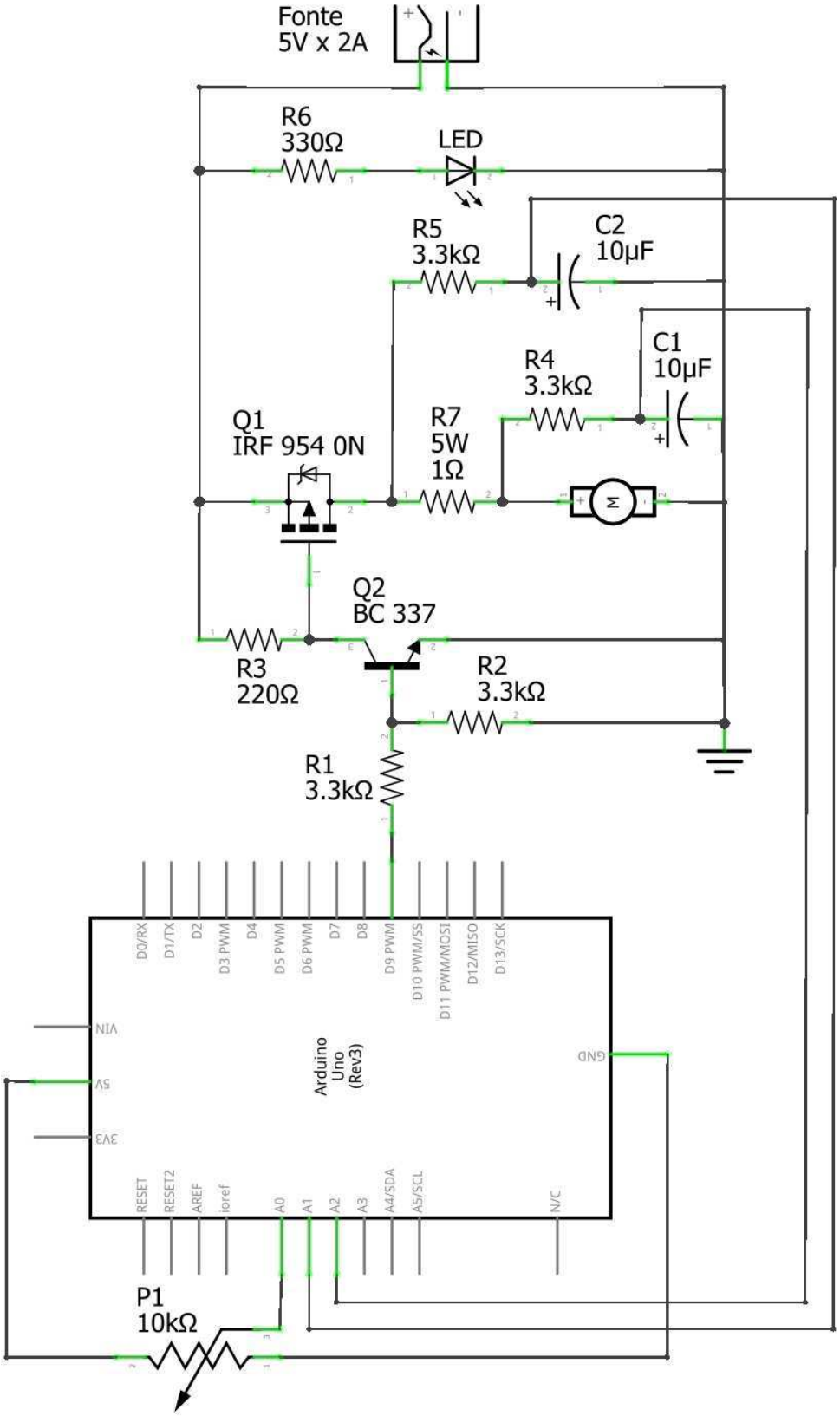
Para embarcar a lógica de controle utilizou-se um Arduino© Uno, um microcontrolador com entradas e saídas digitais e analógicas. Com programação baseada na linguagem C, possui uma interface simples, mas de grande confiabilidade (ARDUINO, 2010), tendo integração com o MATLAB©/Simulink© para monitoramento de dados e geração de código automático. O Arduino© é muito utilizado em ambientes didáticos por sua simplicidade, podendo ser facilmente incorporado a um projeto de automação. No aero pêndulo tem como principal funcionalidade receber todos os sinais de sensores e determinar qual o sinal deve ser enviado ao atuador. Para que isso aconteça foram implementados controladores dos tipos P, PD e PID.

3.3.4 Circuito Eletrônico

Na Figura 9 está representado o esquemático elétrico do sistema, onde:
R1 = R2 = R4 = R5 = 3,3 kΩ
R3 = 220Ω
R6 = 330Ω
R7 = 1 Ω x 5 W
P1 = 10 kΩ
C1 = C2 = 10µF
Q1 = BC 337
Q2 = IRF 954 0N

Pino Analógico A0 - Potenciômetro
Pino Analógico A1 - Tensão do sinal PWM
Pino Analógico A2 - Tensão no motor CC
Pino Digital 9 - Sinal PWM gerado

Figura 9 – Esquemático com conexões dos pinos do Arduino© utilizado



Fonte: Elaboração Própria

4 Modelagem Dinâmica

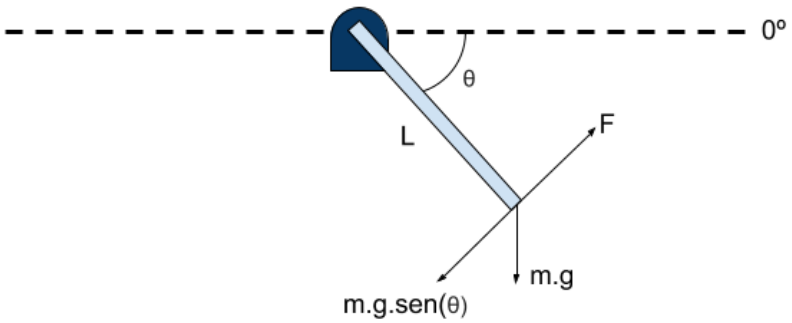
Neste capítulo são apresentadas as modelagens para o aero pêndulo com 1 grau de liberdade, versão construída e implementada, e também para o aero pêndulo com 2 graus de liberdade utilizando a Equação de Lagrange.

4.1 Aero Pêndulo com 1 Grau de Liberdade

O diagrama de forças do aero pêndulo com 1 grau de liberdade é mostrado na Figura 10. Após a aplicação de uma tensão no motor, a força gerada impulsiona o eixo para cima. Como descrito anteriormente, o objetivo do projeto é especificar uma posição angular para o eixo e mantê-lo nessa posição mesmo que ocorram perturbações. Ao controlar a tensão enviada para o motor é possível ajustar a posição angular do eixo, logo, a função de transferência que relaciona posição angular com tensão é de suma importância.

No diagrama temos um ponto fixo localizado na parte superior e a marcação para o horizonte adotado. Foi considerado que a massa está localizada na extremidade do braço, marcado pelo círculo azul, onde também está localizada a hélice e motor que geram a força F. O braço L foi considerado com massa desprezível (MIT, 2016).

Figura 10 – Diagrama de forças do aero pêndulo com um grau de liberdade



Fonte: Elaboração Própria.

De acordo com as Leis de Newton e Momento Angular, a equação do movimento para o aero pêndulo é a seguinte (JOB; JOSE, 2015):

$$J \cdot \frac{d^2\theta}{dt^2} + c \cdot \frac{d\theta}{dt} + m \cdot g \cdot L \cdot sen(\theta) = F \quad (1)$$

Onde,

m = Massa do aero pêndulo (kg)

L = Comprimento do eixo (m)
θ = Posição angular (rad)
c = Coeficiente de amortecimento viscoso (Nms/rad)
J = Momento de Inércia (kgm2)
g = Aceleração da gravidade (m/s2)
F = Força gerada pelo conjunto motor e hélice (N)

Para esse tipo de sistema deve-se considerar que existe uma força contrária ao movimento do aero pêndulo que é gerada pelo ar, ou seja, essa força é proporcional à velocidade do corpo. Na equação (1) o Coeficiente de Amortecimento Viscoso (c) representa o fator de proporcionalidade dessa força de atrito do ar (MOHAMMADBAGHERI; YAGHOOBI, 2011).

O Momento de Inércia (J) mede a forma em que a massa do sistema está distribuída em torno de um eixo, ou seja, ao realizar um movimento angular o corpo tem a tendência de permanecer no estado atual devido a sua massa (ZUBEN, 2010). Para realizar a medição dessa variável usamos a equação (2),

$$J = m \cdot L^2 \qquad (2)$$

Linearizando o sistema do aero pêndulo em torno do ponto de operação que é o horizonte (0°), a aproximação para pequenos ângulos pode ser aplicada na equação (1),

$$J \cdot \frac{d^2\theta}{dt^2} + c \cdot \frac{d\theta}{dt} + m \cdot g \cdot L \cdot \theta = F \quad (3)$$

Assim, a função de transferência que relaciona a posição angular e a força produzida pelo aero pêndulo é dada pela equação (4),

$$\frac{\theta(s)}{F(s)} = \frac{1/J}{s^2 + \frac{c}{J} \cdot s + \frac{m \cdot g \cdot L}{J}} \qquad (4)$$

Nesse contexto, medir a força gerada pelo aero pêndulo se torna inviável, pois seria necessária a instalação de um dinamômetro, além disso, a variável controlada no sistema é a tensão enviada para o motor em questão. Logo, torna-se necessário realizar a caracterização do motor, medindo a força gerada (F) pela tensão aplicada (V) e obtendo um fator de proporcionalidade denominado γ (gamma) (JOB; JOSE, 2015).

$$F(s) = \gamma \cdot V(s) \quad (5)$$

Substituindo a equação (2) e o fator de proporcionalidade da equação (5) na função de transferência do sistema, equação (4), obtida anteriormente, tem-se uma nova função de transferência que relaciona tensão e ângulo.

$$\frac{\theta(s)}{V(s)} = \gamma \cdot \frac{1/m \cdot L^2}{s^2 + \frac{c}{m \cdot L^2} \cdot s + \frac{g}{L}} \qquad (6)$$

Os parâmetros do modelo foram medidos com auxílio de trena para medir o comprimento do braço e uma balança para determinar o peso da planta e também o fator de proporcionalidade entre tensão e força para o motor. O valor para o Coeficiente de Amortecimento Viscoso (c) está intimamente relacionado com a viscosidade do fluido, nesse caso o ar, logo possui um valor definido.

Para determinar o γ (gamma) foi realizado um experimento com balança de precisão e uma massa conhecida, onde ao medir o quanto de massa foi levantada pelo aero pêndulo e com a tensão necessária para isso, consegue-se obter um valor aproximado usando vários pontos de medição e assim obtendo uma equação de primeiro grau, como visto na equação (5).

L = 0,18 m
m = 0,015 kg
g = 9,81 m/s^2
c = 0,0076 Nms/rad
γ = 0,0275

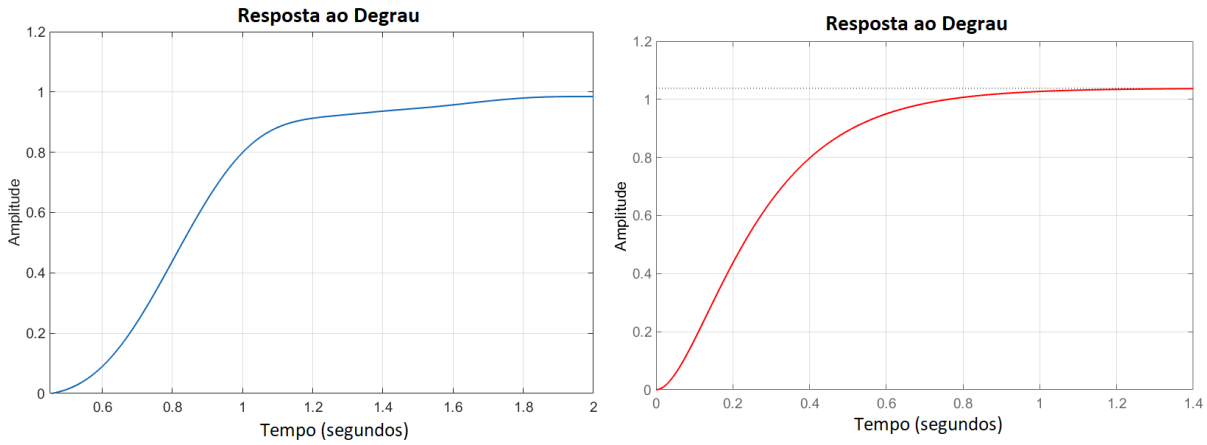
Na Figura 11 estão representadas as respostas ao degrau em malha aberta para as funções de transferência presentes nas equações (6) e do modelo real em malha aberta, aplicando um degrau de tensão na entrada.

Para realizar a medição do ângulo, usando a tensão medida no potenciômetro pelo Arduino©, usamos a equação (7), onde, 202 equivale ao valor medido quando o braço está na posição 0°. Em seguida é calculado o erro entre a referência escolhida e o ângulo medido, que então é enviado para o controlador. O valor de saída do controlador é saturado, entre 0 e 255, pois são os valores de *duty cycle* aceitáveis para o sinal PWM.

4.2 Aero Pêndulo com 2 Graus de Liberdade

Seguindo os mesmos objetivos do sistema anterior, o aero pêndulo com 2 graus de liberdade foi pensado adicionando mais um eixo de rotação. Para que seja possível realizar

Figura 11 – Resposta ao degrau em malha aberta, em azul modelo real, em vermelho modelo simulado



Fonte: Elaboração Própria.

$$Angle = \frac{202 - V_{pot}}{4} \quad (7)$$

os movimentos, foi adicionado mais um conjunto motor e hélice e a disposição dos mesmos se dá nas extremidades do segundo eixo de tamanho $2l_2$.

Para a modelagem foi utilizado a Equação de Lagrange (DEYST, 2003) para obtenção das equações de movimento e do modelo em espaço de estados. Na figura 12 temos o diagrama de forças desse sistema, onde os ângulos de rotação θ e φ são movimentados pela atuação dos dois motores m_1 e m_2. Ao controlar cada um dos motores conseguimos alcançar uma posição desejada em cada um dos eixos de rotação.

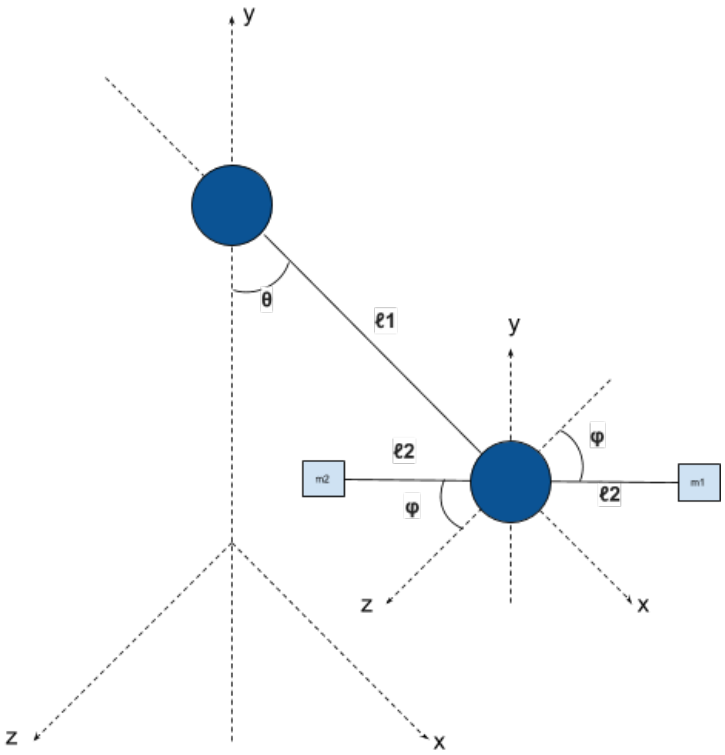
No diagrama temos dois pontos fixos, em azul escuro. A massa de todo o sistema está concentrada na extremidade do segundo eixo de rotação, onde estão localizados os conjuntos de hélice e motor, semelhantes aos utilizados no sistema anterior. Logo, os braços l_1 e $2l_2$, foram considerados com massa desprezível.

Inicialmente, analisando o diagrama de forças, percebemos que o movimento do ângulo θ influencia nas posições x e y, já o movimento do ângulo φ influencia nas posições y e z. Logo, podemos escrever a posição e velocidade em cada eixo em função dos ângulos.

Como o movimento do eixo x é o mesmo para os dois motores, só temos uma equação, e, por serem simétricos, os eixos y e z mudam apenas o sinal do movimento do ângulo φ. Então, derivando as equações de posição, obtemos as equações de velocidade.

Sabendo que o Lagrangiano L é resultado da diferença entre a energia cinética (KE) e a energia potencial (PE), utilizando as equações de posição e velocidade obtidas anteriormente podemos calcular os mesmos e com isso conseguimos obter a equação de

Figura 12 – Diagrama de forças do aero pêndulo com dois graus de liberdade



Fonte: Elaboração Própria.

$$x = l_1 \cdot sen(\theta) \quad (8)$$

$$y_1 = l_2 \cdot sen(\varphi) - l_1 \cdot cos(\theta) \quad (9)$$

$$y_2 = -l_2 \cdot sen(\varphi) - l_1 \cdot cos(\theta) \quad (10)$$

$$z_1 = -l_2 \cdot cos(\varphi) \quad (11)$$

$$z_2 = l_2 \cdot cos(\varphi) \quad (12)$$

movimento do sistema, diferente do método de Newton que requer a equação de aceleração para todas as direções.

$$\dot{x} = l_1 \cdot cos(\theta) \cdot \dot{\theta} \quad (13)$$

$$\dot{y}_1 = l_1 \cdot sen(\theta) \cdot \dot{\theta} + l_2 \cdot cos(\varphi) \cdot \dot{\varphi} \quad (14)$$

$$\dot{y}_2 = l_1 \cdot sen(\theta) \cdot \dot{\theta} - l_2 \cdot cos(\varphi) \cdot \dot{\varphi} \quad (15)$$

$$\dot{z_1} = l_2 \cdot sen(\varphi) \cdot \dot{\varphi} \quad (16)$$

$$\dot{z_2} = -l_2 \cdot sen(\varphi) \cdot \dot{\varphi} \quad (17)$$

$$PE = m_1 \cdot g \cdot y_1 + m_2 \cdot g \cdot y_2 = \\ = g \cdot l_2 \cdot sen(\varphi) \cdot (m_1 - m_2) - g \cdot l_1 \cdot cos(\theta) \cdot (m_1 + m_2) \quad (18)$$

$$KE = \frac{1}{2} \cdot m_1 \cdot (\dot{x}^2 + \dot{y}_1{}^2 + \dot{z}_1{}^2) + \frac{1}{2} \cdot m_2 \cdot (\dot{x}^2 + \dot{y}_2{}^2 + \dot{z}_2{}^2) \\ = (m_1 + m_2) \cdot (\frac{l_1{}^2}{2} \cdot \dot{\theta}^2 + \frac{l_2{}^2}{2} \cdot \dot{\varphi}^2) + \\ (m_1 - m_2)(l_1 \cdot l_2 \cdot \dot{\theta} \cdot \dot{\varphi} \cdot sen(\theta) \cdot cos(\varphi)) \quad (19)$$

$$L = KE - PE \\ = (m_1 + m_2) \cdot (\frac{l_1{}^2}{2} \cdot \dot{\theta}^2 + \frac{l_2{}^2}{2} \cdot \dot{\varphi}^2 + g \cdot l_1 \cdot cos(\theta)) + \\ (m_1 - m_2) \cdot (l_1 \cdot l_2 \cdot \dot{\theta} \cdot \dot{\varphi} \cdot sen(\varphi) \cdot cos(\theta) - g \cdot l_2 \cdot sen(\varphi)) \quad (20)$$

Substituindo L na Equação de Lagrange e igualando à força que atua na direção em que foi calculada, sendo F1 e F2 as forças geradas pelo par de motores e hélices. Assim, obtemos as equações de movimento do sistema para cada ângulo de rotação.

Equações de movimento para o ângulo θ:

$$\frac{\partial L}{\partial \theta} = (m_1 + m_2)(-g \cdot l_1 \cdot sen(\theta)) + (m_1 - m_2)(l_1 \cdot l_2 \cdot \dot{\theta} \cdot \dot{\varphi} \cdot cos(\theta) \cdot cos(\varphi)) \quad (21)$$

$$\frac{\partial L}{\partial \dot{\theta}} = (m_1 + m_2)({l_1}^2 \cdot \dot{\theta}) + (m_1 - m_2)(l_1 \cdot l_2 \cdot \dot{\varphi} \cdot sen(\theta) \cdot cos(\varphi)) \quad (22)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) = (m_1 + m_2)({l_1}^2 \cdot \ddot{\theta}) + (m_1 - m_2)(l_1 \cdot l_2 \cdot \ddot{\varphi} \cdot sen(\theta) \cdot cos(\varphi) + l_1 \cdot l_2 \cdot \dot{\varphi} \cdot \dot{\theta} \cdot cos(\theta) \cdot cos(\varphi) - l_1 \cdot l_2 \cdot \dot{\varphi}^2 \cdot sen(\theta) \cdot sen(\varphi)) \quad (23)$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} &= (m_1 + m_2)({l_1}^2 \cdot \ddot{\theta} + g \cdot l_1 \cdot sen(\theta)) \\ &+ (m_1 - m_2)(l_1 \cdot l_2 \cdot \dot{\varphi} \cdot sen(\theta) \cdot cos(\varphi)) \\ &- (m_1 - m_2)(l_1 \cdot l_2 \cdot \dot{\varphi}^2 \cdot sen(\theta) \cdot sen(\varphi))\end{aligned} \quad (24)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) - \frac{\partial L}{\partial \theta} = (F_1 + F_2) \cdot sen(\varphi) \quad (25)$$

Equações de movimento para o ângulo ϕ:

$$\frac{\partial L}{\partial \varphi} = -(m_1 - m_2)(l_1 \cdot l_2 \cdot \dot{\theta} \cdot \dot{\varphi} \cdot sen(\theta) \cdot sen(\varphi) + g \cdot l_2 \cdot cos(\varphi)) \quad (26)$$

$$\frac{\partial L}{\partial \dot{\varphi}} = (m_1 + m_2)({l_2}^2 \cdot \dot{\varphi}) + (m_1 - m_2)(l_1 \cdot l_2 \cdot \dot{\theta} \cdot sen(\theta) \cdot cos(\varphi)) \quad (27)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\varphi}}\right) = (m_1 + m_2)({l_2}^2 \cdot \ddot{\varphi}) + (m_1 - m_2)(l_1 \cdot l_2 \cdot \ddot{\theta} \cdot sen(\theta) \cdot cos(\varphi) + l_1 \cdot l_2 \cdot \dot{\theta}^2 \cdot cos(\theta) \cdot cos(\varphi) - l_1 \cdot l_2 \cdot \dot{\theta} \cdot \dot{\varphi} \cdot sen(\theta) \cdot sen(\varphi)) \quad (28)$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \varphi}\right) - \frac{\partial L}{\partial \varphi} &= (m_1 + m_2)({l_2}^2 \cdot \ddot{\varphi}) + (m_1 - m_2)(l_1 \cdot l_2 \cdot \ddot{\theta} \cdot sen(\theta) \cdot cos(\varphi) \\ &+ l_1 \cdot l_2 \cdot \dot{\theta}^2 \cdot cos(\theta) \cdot cos(\varphi) + g \cdot l_2 \cdot cos(\varphi))\end{aligned} \quad (29)$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \varphi}\right) - \frac{\partial L}{\partial \varphi} = (F_1 - F_2) \cdot cos(\varphi) \quad (30)$$

Semelhante ao que foi feito na modelagem dinâmica do aero pêndulo com um grau de liberdade, a força gerada se torna inviável de ser utilizada na modelagem. Logo, por utilizar o mesmo conjunto de motor e hélice, aplicamos o fator de proporcionalidade γ (gamma) da força gerada pela tensão enviada aos motores.

$l1 = 0{,}18\ m$
$l2 = 0{,}06\ m$
$\mathrm{m1} = \mathrm{m2} = 0{,}015\ kg$
$\mathrm{g} = 9{,}81\ m/s^2$

$\gamma = 0{,}0275$

5 Controle PID

Os dois problemas fundamentais de controle podem ser caracterizados como rastreamento de referência e rejeição de perturbações, para ambos os casos normalmente é projetado um controlador que atinja tais objetivos (BATISTA, 2014). Para avaliar a performance do sistema de controle, existem uma série de métricas, entre elas, erro estacionário, tempo de subida e sobressinal.

Na plataforma desenvolvida utiliza-se um dos modelos de controladores mais difundidos e que já está implementado em um bloco do Simulink©, o controlador PID. No MATLAB© existe uma ferramenta de sintonia para controladores P, PI, PD e PID que será abordada nas próximas seções.

5.1 Formulação

Os Controladores P, PD e PID determinam o sinal que deve ser enviado para a planta de acordo com o erro entre a saída do sistema e a referência escolhida. No PID o erro passa por três etapas, um ganho proporcional (Kp), um integral (Ki) e um derivativo (Kd). Esses ganhos são responsáveis por moldar a dinâmica da saída para uma variação de referência ou uma acomodação a perturbações.

Os controladores P, PD e PID tem como função de transferência as equações (31), (32) e (33) descritas abaixo, respectivamente,

$$Gp(s) = Kp \quad (31),$$

$$Gpd(s) = Kp + Kd \cdot s \quad (32),$$

$$Gpid(s) = Kp + Kd \cdot s + \frac{Ki}{s} = \frac{Kd \cdot s^2 + Kp \cdot s + Ki}{s} \quad (33)$$

5.2 Sintonia

Quando controladores são utilizados deve-se procurar por valores de ganhos que satisfaça a especificação de desempenho desejada em malha fechada, esse processo é chamado de sintonia do controlador.

Tendo em mãos a função de transferência de malha fechada do sistema, pode-se então usar métodos matemáticos como Ziegler-Nichols, SIMC, entre outros, ou então recorrer à softwares que calculam esses valores por meio de simulações.

5.2.1 *PID Tuner Toolbox*

A *toolbox* chamada *'PID Tuner'* permite calcular, a partir da função de transferência de malha aberta, os valores dos ganhos para um controlador do tipo selecionado pelo usuário, podendo ser P, PI, PD, PID e outros.

Ajustando o tempo de resposta e a robustez do sinal de saída, pode-se obter diferentes valores para Kp, Ki e Kd e transferir esses valores diretamente para o controlador utilizado no Simulink©.

Além disso, são gerados gráficos que mostram a resposta da função de transferência em malha fechada e também são fornecidos os valores dos parâmetros de desempenho para o controlador obtido.

5.2.2 *Model-In-the-Loop*

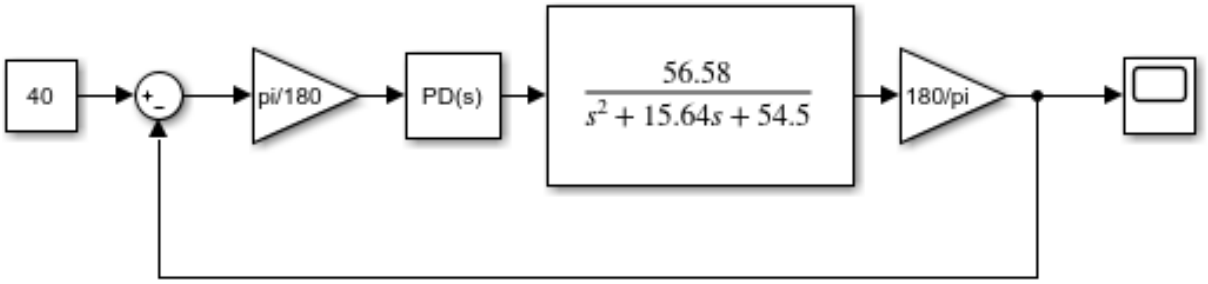
De posse das equações de malha aberta e das equações do controlador, pode-se então obter o modelo de equações de malha fechada para o sistema do aero pêndulo.

Na Figura 13 temos o diagrama de malha fechada do sistema e na Tabela 1, os ganhos obtidos pelo *PID Tuner Toolbox* para uma escolha conservadora de robustez. São ilustrados na Figura 14 os gráficos da saída do sistema para uma variação em degrau da referência de 0° até 40°. Foram utilizados controladores P, PD e PID.

Tabela 1 – Ganhos sintonizados do controlador

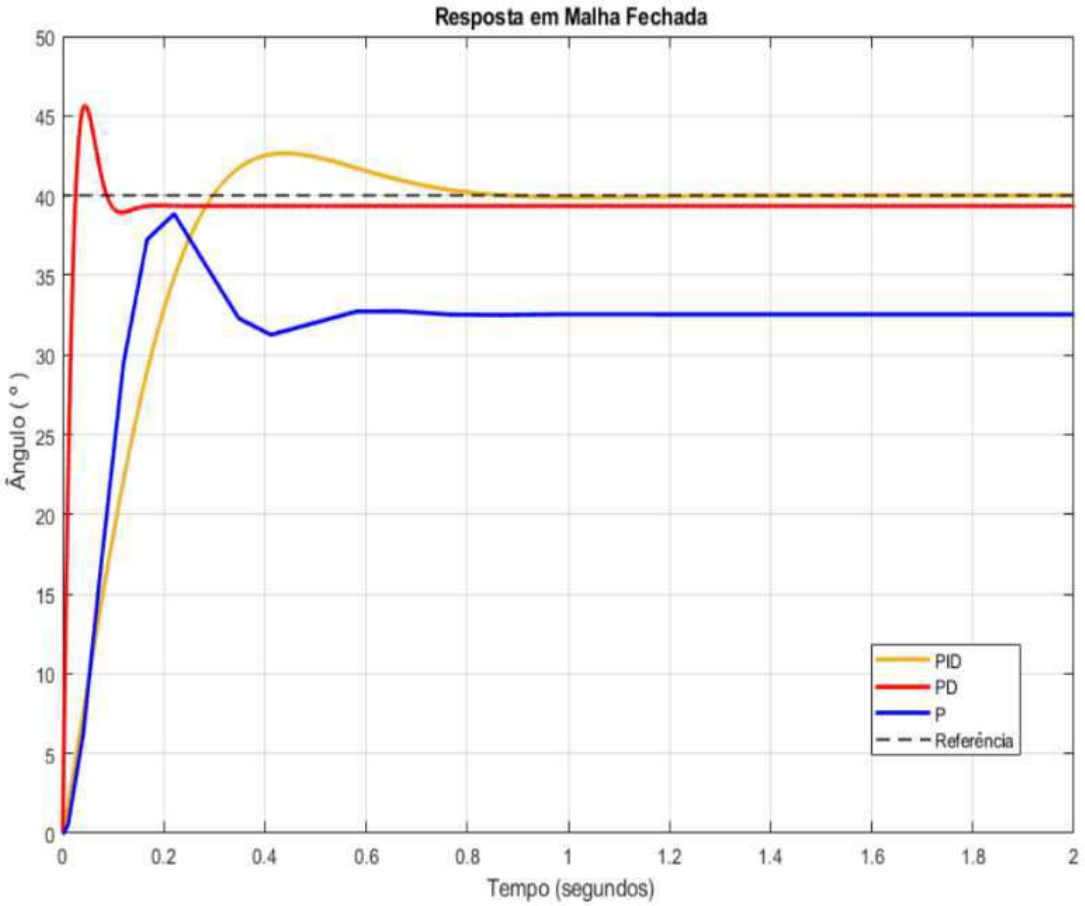
Controlador	**P**	**PD**	**PID**
Kp	4,205	57,931	1,9
Kd	-	1,0558	0,0788
Ki	-	-	9,694
N	-	8436,8264	843,7

Figura 13 – Diagrama de malha fechada



Fonte: Elaboração Própria.

Figura 14 – Resposta de malha fechada



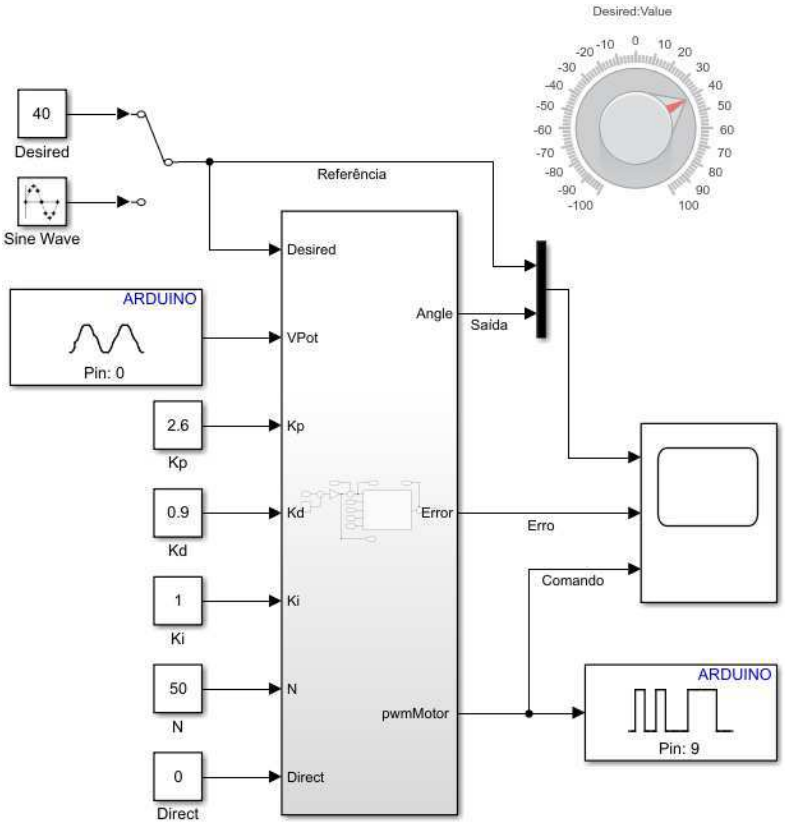
Fonte: Elaboração Própria.

6 Resultados Obtidos

Após todos os passos de simulação e design realizados, inicia-se o processo de implementação descrito no diagrama do *V-Model*. Para isso utiliza-se o Simulink© para gerar automaticamente o código do controlador que será carregado no Arduino©.

O modelo desenvolvido no Simulink© está configurado para executar em modo *'External'* permitindo a comunicação em tempo real com o Arduino© Uno, tornando possível então traçar os gráficos com dados de erro, ângulo medido e do sinal enviado para o atuador. É mostrado nas figuras 15 e 16 o modelo implementado.

Figura 15 – Representação do modelo do aero pêndulo no Simulink©

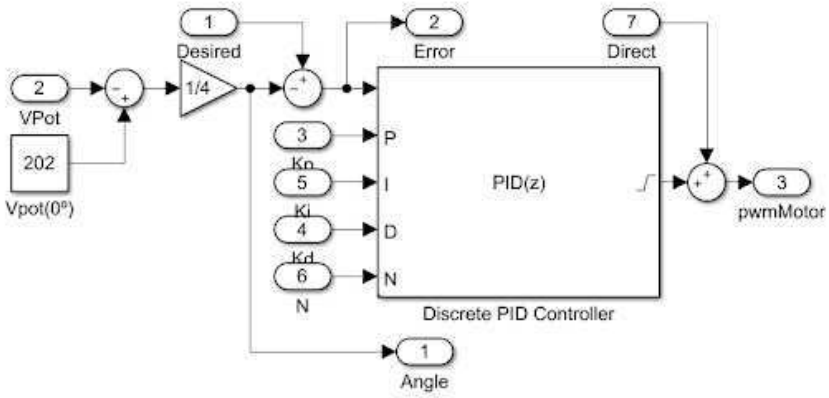


Fonte: Elaboração Própria.

Os valores de ganhos para o PID aplicado na planta real são, Kp = 2,6, Kd = 0,9, Ki = 1 e N = 50. Na Figura 18 são mostrados os gráficos gerados, respectivamente, ângulo e referência, erro e comando para o motor. Para esse teste foi considerada como posição angular inicial -35° e a referência desejada foi 35°.

Na Figura 17 podemos analisar as respostas do modelo e da planta real em malha fechada. Para esse teste foi considerada a referência saindo de 0° até 20°. Percebe-se que

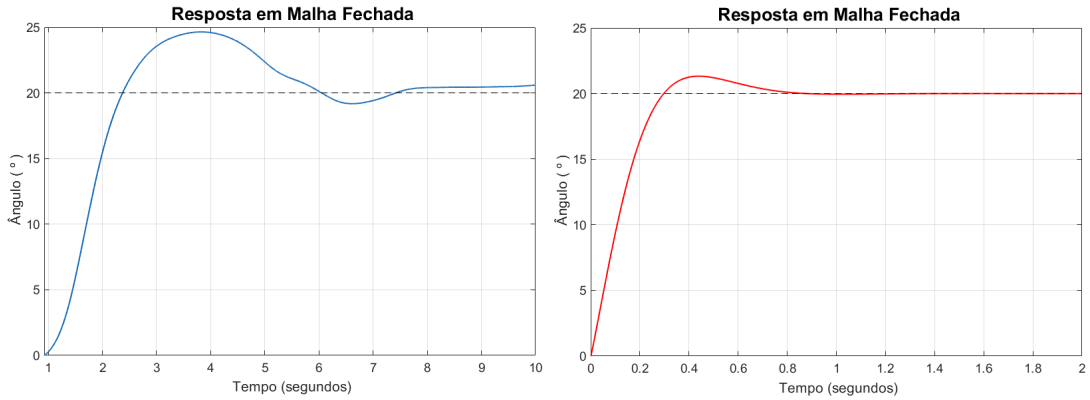
Figura 16 – Representação da parte interna do bloco central do modelo



Fonte: Elaboração Própria.

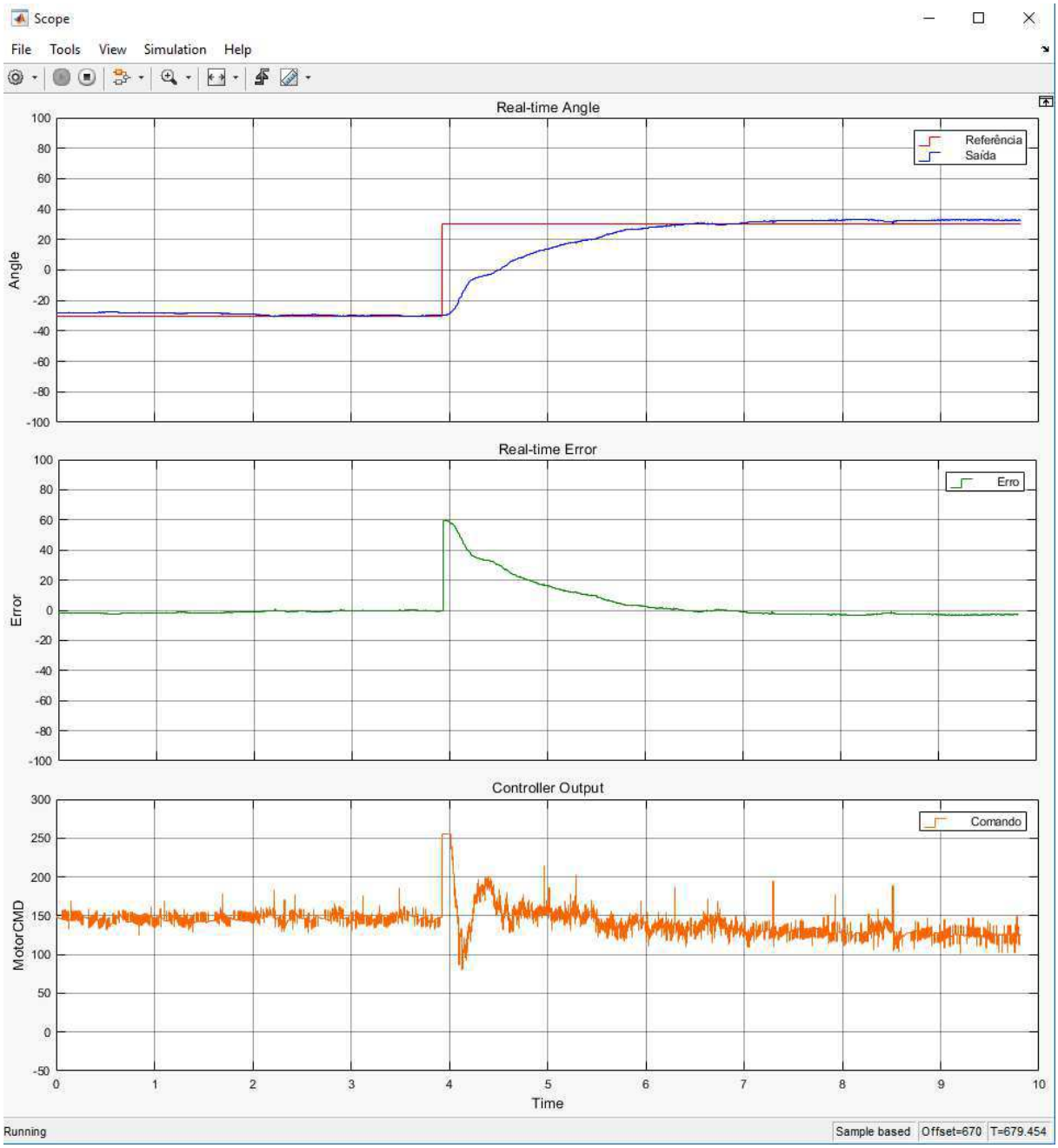
as dinâmicas de resposta são praticamente iguais, diferenciando pelo tempo de subida e acomodação, que por questões de atrito mecânico entre as peças, atrito com relação ao ar e atrasos de comunicação entre Arduino©, Simulink© e motor na planta real geram diferenças com relação ao modelo matematicamente simulado.

Figura 17 – Comparação entre resposta real, em azul, e simulado, em vermelho, para o controlador PID



Fonte: Elaboração Própria.

Figura 18 – Gráficos gerados experimentalmente



Fonte: Elaboração Própria.

7 Considerações Finais

Conclui-se então que esse estudo possibilitou o desenvolvimento de uma plataforma didática para ensino de modelagem e sintonia de sistemas de controle, utilizando o software MATLAB©/Simulink© como ferramenta de codificação e simulação e o aero pêndulo como planta a ser controlada. Devido a fácil aquisição dos componentes utilizados torna-se acessível à diversas aplicações de ensino.

Os resultados obtidos mostram que os controladores P, PD e PID alcançaram os requisitos especificados pois tornaram possível o controle da planta. Além disso, o processo de sintonia foi facilitado pelo *'PID Tuner Toolbox'* e a geração de código automático permite a implementação do controlador sem a necessidade de conhecer a linguagem nativa de programação do microcontrolador, diminuindo o tempo de implementação.

Podem ser aplicados ao aero pêndulo outras modelagens e outros métodos de sintonia, obtendo assim resultados diferentes, porém, podem ser utilizadas as mesmas ferramentas e a mesma plataforma deste trabalho para implementação do controlador, justificando assim o caráter didático da plataforma.

Referências Bibliográficas

AARENSTRUP, R. *Managing Model-Based Design*. [S.l.]: MathWorks, Inc, 2015. Citado 3 vezes nas páginas 2, 4 e 5.

ARDUINO. *Arduino Uno Rev3*. 2010. Disponível em: <https://store.arduino.cc/usa/arduino-uno-rev3>. Citado na página 12.

BARROS, A. et al. Construção e projeto de controle de um aero pêndulo utilizando *Model-Based Design*. *Congresso Brasileiro de Educação em Engenharia*, 2019. Citado na página 1.

BATISTA, L. Estudo comparativo de técnicas de sintonia de controladores pid para sistemas de primeira ordem com atraso. *ABCM Symposium Series in Mechatronics*, v. 6, 2014. Citado na página 22.

CHAUHAN, K. Why is Model-Based Design important in Embedded Systems. *<https://www.einfochips.com/blog/why-is-model-based-design-important-in-embedded-systems>*, 2018. Citado na página 4.

DEYST, J. J. *Lagrange's Equations*. [S.l.]: Massachusetts Institute of Technology, MIT, 2003. Citado na página 17.

ENIKOV, E. T.; CAMPA, G. Mechatronic aeropendulum: Demonstration of linear and nonlinear feedback control principles with matlab/simulink real-time windows target. *IEEE Transactions on Education*, v. 55, n. 4, 2012. Citado na página 1.

JOB, M. M.; JOSE, P. S. H. Modeling and control of mechatronic aeropendulum. *IEEE Sponsored 2nd International Conference on Innovations in Information Embedded and Communication Systems*, 2015. Citado 3 vezes nas páginas 9, 14 e 15.

LIN, J. *Measuring Return on Investment of Model-Based Design*. [S.l.]: MathWorks, Inc, 2011. Citado na página 4.

MATHWORKS. *Simulink Support Package for Arduino Hardware*. 2016. Disponível em: <https://www.mathworks.com/help/supportpkg/arduino/>. Citado 2 vezes nas páginas 2 e 7.

MIT. *Courses edX: Introduction to Control System Design – A First Look*. 2016. Disponível em: <https://edx.org/course/introduction-control-system-design-first-mitx-6-302-0x/>. Citado na página 14.

MOHAMMADBAGHERI, A.; YAGHOOBI, M. A new approach to control a driven pendulum with pid method. *UKSim 13th International Conference on Modelling and Simulation*, 2011. Citado na página 15.

SEEEDSTUDIO. *Grove - Starter Kit for Arduino*. 2012. Disponível em: <https://www.seeedstudio.com/Grove-Starter-Kit-for-Arduino-p-1855.html>. Citado na página 8.

ZUBEN, F. J. V. *Modelagem de Sistemas Dinâmicos Contínuos no Tempo*. [S.l.]: DCA/FEEC/Unicamp, 2010. 27-30 p. Citado na página 15.