DAS Departamento de Automação e Sistemas
CTC Centro Tecnológico
UFSC Universidade Federal de Santa Catarina

Controle de um robô paralelo de seis graus de liberdade para uso em simuladores de voo

Relatório submetido à Universidade Federal de Santa Catarina

como requisito para a aprovação da disciplina:

DAS 5511: Projeto de Fim de Curso

Georgiy Carlos Tanca Nazarov

Florianópolis, Dezembro de 2018

Controle de um robô paralelo de seis graus de liberdade para uso em simuladores de voo

Georgiy Carlos Tanca Nazarov

Esta monografia foi julgada no contexto da disciplina

DAS 5511: Projeto de Fim de Curso

e aprovada na sua forma final pelo

Curso de Engenharia de Controle e Automação

Prof. Henrique Simas

Assinatura do Orientador

Banca Examinadora:

Prof. Mauricio Becerra Vargas

Orientador na Instituição

Prof. Henrique Simas

Orientador no Curso

Prof. Ricardo Rabelo

Responsável pela disciplina

Prof. Ubirajara Franco Moreno

Avaliador

Thiago Maurici Espindola

Debatedor

Alex Amadeu Cani

Debatedor

Agradecimentos

Agradeço ao Prof. Mauricio pela oportunidade de realizar meu PFC no Grupo de Automação e Sistema Integrados - GASI, e pelo continuo apoio que me deu durante o desenvolvimento do projeto, sem o qual não poderia ter realizado este trabalho.

Agradeço a minha família que me apoio durante toda a minha graduação no Brasil.

Agradeço a UNESP - Sorocaba e a todos seus funcionários, docentes e discentes que me ajudaram no desenvolvimento do meu PFC.

Agradeço a meu orientador na UFSC pela paciência e ajuda que prestou durante o PFC.

Agradeço a os professores, técnicos administrativos e todos os demais trabalhadores da UFSC que ao longo da minha graduação me ajudaram e se esmeraram em cumprir suas funções para fazer a UFSC uma das melhores intuições de ensino superior no Brasil e na América Latina da qual tenho orgulho de fazer parte.

Agradeço a todos meus amigos do Brasil e do mundo a fora.

Resumo

Na atualidade, simuladores de voo fazem parte do processo de treinamento de novos pilotos na industria da aviação. No GASI, o Prof. Mauricio Becerra Vargas coordena um projeto para implementar um simulador de voo na UNESP/Sorocaba. O simulador possui uma plataforma de Stewart para simular movimentos no corpo do piloto, tal plataforma precisa de um sistema de controle que garanta que o robô se movimente de forma correta. Este documento relata os processo realizado para implementar 4 tipo de controladores diferentes para realizar esta tarefa. Os controladores diferem um dos outros, pelo método usado para projetar eles, este foram: sintonização PID, lugar das raízes, alocação de polo e LQR. Também foram realizados procedimentos necessários para implementar corretamente estes controladores, como modelagem do sistema e filtragem da medida. Os resultados destes controladores são apresentados, avaliados e comparados neste relatório.

Palavras-chave: Plataforma de Stewart; Simuladores de Voo; Controle PID; Lugar das Raízes; Alocação de Polos; Regulador Linear Quadrático; Filtro de Medida; Identificação de Sistemas; Controle Robótico.

Abstract

Nowadays, flight simulators are part of the training process for new pilots in the aviation industry. In the GASI, Prof. Mauricio Becerra Vargas coordinates a project to implement a flight simulator in the UNESP/Sorocaba. The flight simulator possess a Stewart platform to simulate movements in the body of the pilot, this platform requires a control system to guarantee that the robot moves in the correct manner. This document reports the process made to implement 4 types of different controllers to accomplish this task. The controllers differ one from another by the method used to design them, these methods were: PID tuning, root locus, pole placement and LQR. Likewise, the necessary proceedings were executed to correctly implement these controllers, like the modeling of the system and filtering of the measurement. The results obtained by these controllers are presented, evaluated and compared in this report.

Keywords: Stewart Platform; Flight Simulators; PID control; Root Locus; Pole Placement; LQR; Measurement Filter; System Identification; Robotic Control.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

Lista de Siglas

ADC - *Analog-to-Digital Converter*

CC - Corrente contínua

DAC - *Digital-to-Analog Converter*

DOF - *Degrees of Freedom*

EDO - Equação Diferencial Ordinária

GASI - Grupo de Automação e Sistemas Integráveis

GDL - Graus de Liberdade

GPS - *Global Positioning System*

IMU - *Inertial Measurement Unit*

INS - *Inertial Navigation System*

LQR - *Linear–Quadratic Regulator*

MIMO - *Multiple Input - Multiple Output*

PID - Proporcional Derivativo Integrativo

PP - *Pole Placement*

PWM - *Pulse Width Modulation*

RL - *Root Locus*

RS-232 – *Recommended Standard 232*

SCADA - *Supervisory Control and Data Acquisition*

SISO - *Single Input - Single Output*

UFSC - Universidade Federal de Santa Catarina

UNESP - Universidade Estadual Paulista "Júlio de Mesquita Filho"

UPUR - Universal Prismatic Universal Rotation Joints

USB - *Universal Serial Bus*

Lista de símbolos

°- Graus

bps - Bits por segundo

cm - Centímetros

dB - Decibéis

Hz - Hertz

m - Metros

mm - Milímetros

m/s - Metros por segundo

rad/s - Radianos por segundo

s - Segundos

t - Tempo

V - Volts

$C(\cdot)$ - cos$(\cdot)$

$S(\cdot)$ - sen$(\cdot)$

$\overset{n}{x}(t)$ - Derivada no tempo de ordem n da função $x(t)$

Sumário

1 Introdução

Os simuladores de voo são ferramentas importantes para a capacitação de pilotos, uma vez que, com a evolução da aviação moderna, o uso de aeronaves cada vez maiores e mais custosas torna os métodos antigos perigosos e pouco eficientes. Neste contexto Doug Stewart publicou, em 1965, seu artigo onde apresentou um mecanismo de 6 GDL controlado por seis motores independentes para uso em simuladores de voo [1]. Diferente de outros simuladores da época, o mecanismo de Stewart era capaz de se movimentar em todas as direções e girar em todos os eixos, para simular o *roll*, *pitch* e *yaw* de uma aeronave, claro esta dentro das próprias limitações físicas do mecanismo. Além disto, na visão de Stewart, o mecanismo poderia ser usado além da sua concepção inicial para outros fins [1].

O mecanismo de Stewart se tratava de um robô paralelo de 6 GDL, no entanto este não é igual ao que atualmente se conhece por "Plataforma de Stewart" como se pode apreciar na figura 1. Além disto previamente ao artigo de Stewart, o Dr. Eric Gough fabricou o que se poderia, ironicamente, denominar como a primeira "Plataforma de Stewart" [2] em 1947. A maquina de Gough, chamada por ele de "equipamento universal" (*universal rig*), era usada para fazer testes de carga em pneus, como se pode observar na figura 2, e esteve em funcionamento até 1999. [2]

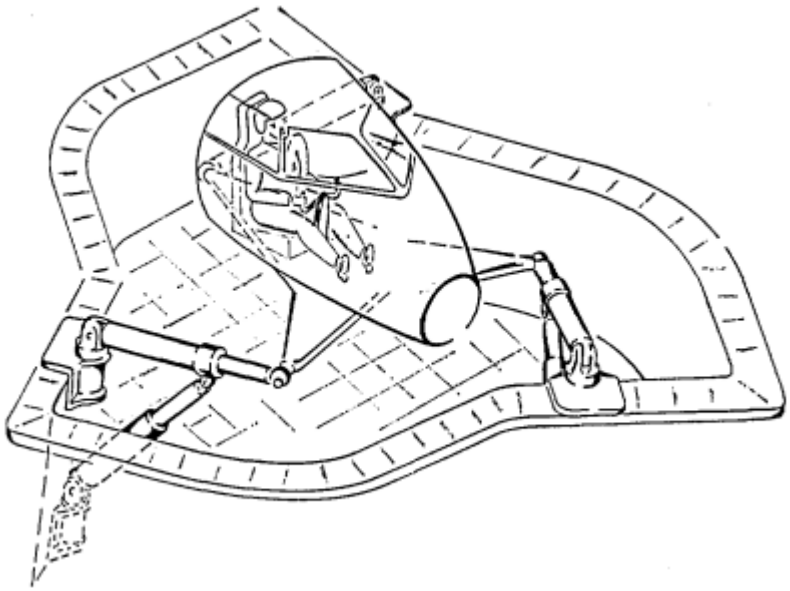


Figura 1 – Mecanismo proposto por D. Stewart [1]

Ainda por cima da invenção de Gough, em 1962, o engenheiro Klaus L. Cappel tinha inicialmente a tarefa de melhorar um sistema de vibração, semelhante à plataforma de Stewart, o que o familiarizou com os robôs paralelos. Nessa mesma década, uma corporação,

o encarregou de construir um simulador de voo para helicópteros, assim nasceu a primeira "Plataforma de Stewart" usada para simuladores de voo. O mecanismo proposto por Cappel tinha a mesma configuração que a construída por Gough como se pode observar na figura 3. Cappel requereu uma patente em 1964 da sua invenção para o uso em simuladores de movimento [10], desconhecendo da existência da maquina de Gough.

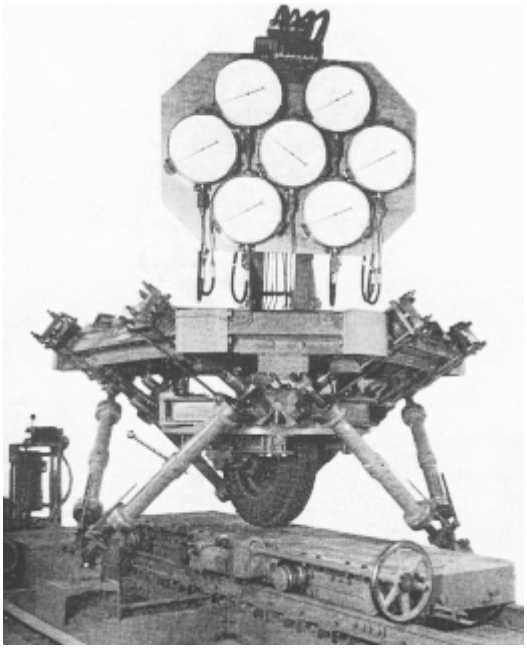


Figura 2 – Maquina construída por E. Gough [2]

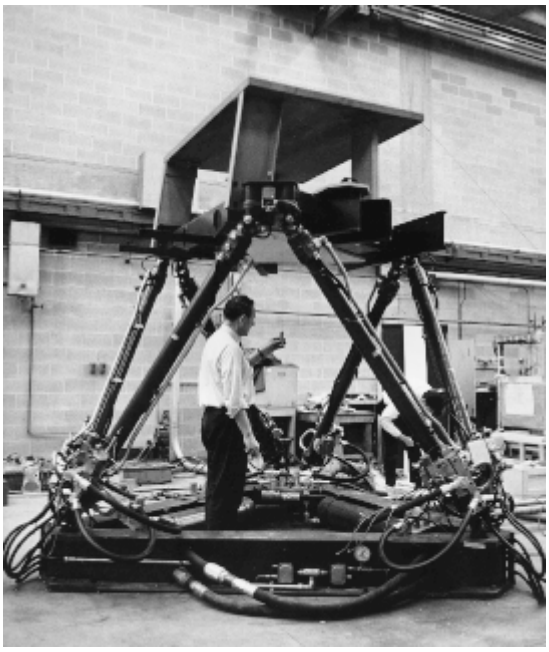


Figura 3 – Simulador de movimento patenteado por K. L. . Cappel [2]

Três pessoas diferentes que, inconscientemente, criaram e/ou propuseram este mesmo tipo de robô paralelo, devido à falta de comunicação entre diferentes pesquisadores

na época. Não se pode negar que todos eles tiveram um papel importante na popularização do que, hoje, é conhecido como a plataforma de Stewart. As previsões de Stewart se tornaram corretas e, apesar de ter se passado mais de meio século, a plataforma de Stewart se tornou uma referencia quando se fala de robôs paralelos e seu uso vai além de simuladores de movimento.

A plataforma de Stewart tem sua popularidade no uso de simuladores de voo devido a que é possível simular com relativa facilidade os movimentos de 6 GDL, ou seja, movimentos nas coordenadas cartesianas e nas coordenadas angulares. Isto é necessário para poder simular as sensações de aceleração e orientação que o corpo de um piloto sentiria ao pilotar uma aeronave real.

No laboratório do Grupo de Automação e Sistemas Integráveis da Universidade Estadual Paulista, o Prof. Mauricio coordena um projeto da sua autoria, para implementar um simulador de voo. Dentro deste projeto é necessário realizar o controle da plataforma de Stewart do simulador de voo, do qual o autor é um dos encarregados.

Este documento, explica as teorias usadas para poder implementar este sistema de controle, os projetos realizado e os resultados obtidos. Como o simulador ainda esta em fase de construção, com alguns elementos sendo desenvolvidos em paralelo e outros ainda não implementados, se decidiu projetar o sistema de controle com diferentes metodologias de projeto, e comparar os resultados.

Os projetos propostos usam a estrutura de controle no espaço das juntas descentralizado, onde os controladores são desenvolvidos usando 4 metodologias diferentes. Estas são: sintonização PID, lugar das raízes, alocação de polos e regulador linear quadrático.

Ressalta-se que, embora os métodos usados sejam diferentes, todos eles são métodos lineares implementados para sistemas SISO, e, por tanto, eles são todos equivalentes, ou seja, são apenas caminhos diferentes usados que podem chegar aos mesmos resultados. No entanto será explicado a razão do uso de métodos diferentes na seção a seguir.

Este documento foi redigido pensando em leitores que já tenham conhecimentos básicos sobre robótica e teoria de controle clássica e moderna. Também é desejável que se tenha familiaridade com o uso do *software* da MathWorks, MATLAB e Simulink, embora isto não seja indispensável para o entendimento da implementação dos controladores.

1.1 Objetivos Gerais do Projeto

O projeto desenvolvido teve como principal objetivo projetar, simular e testar em tempo real um sistema de controle **preliminar**, ou seja, um sistema inicial para o testar o funcionamento do sistema de movimento do simulador do voo desenvolvido no GASI.

O intuito é usar este sistema de controle como uma base para poder comparar

posteriormente sistemas de controle mais complexos. Sendo assim este os métodos de controles usados durante o projeto são métodos amplamente conhecidos nas teorias de controle clássica e moderna.

O objetivo secundário era avaliar a dificuldade de usar diferentes métodos para projetar o sistema de controle, assim como obter conhecimento sobre os problemas e dificuldades que se encontraram para realizar este sistema de controle.

Assim, este documento pode ser usado como uma referência para o desenvolvimento futuro de outros sistemas de controle.

1.2 Estrutura do Documento

O documento é dividido em 8 capítulos, contando este capítulo. Os capítulos são dividido em seções, conforme seja necessário, para facilitar o entendimento dos leitores.

No capítulo 2 se apresenta o funcionamento do simulador de voo, respectivamente em cada um dos seus elementos, teóricos e reais. Se dá uma enfase ao sistema ao qual os controladores, que serão projetados pelo aluno, fazem parte. Também se faz uma definição formal do problema e dos requisitos que o sistema de movimento do simulador de voo deve cumprir.

No capitulo 3 e 4 são apresentados os embasamentos teóricos que foram usados para pode implementar este sistema de controle. Especificamente no capítulo 3 se apresenta os métodos usados para realizar realizar o projeto dos controladores, como as metodologias e estruturas usadas, ou que poderiam ser usadas. Já no capítulo 4 é dedicado a explicar métodos adicionais que foram necessários para poder implementar os controladores, estes são filtro de medida, identificação de sistemas e absorvedor de estados.

O capítulo 5 explica o processo realizado para projetar os diversos elementos que farão parte do sistema de controle, a maioria destes forma feitos usando o MATLAB e/ou Simulink. Assim como a obtenção de um modelo no qual basear o projeto de de controle.

O capítulo 6 é dedicado a explicar como o sistema de controle projetado é implementado para funcionar em tempo real com a plataforma de Stewart se movimentando. Isto é feito usando o *software* da dSPACE, chamado de ControlDesk, portanto se explica o funcionamento básico deste.

Se dedica exclusivamente o capítulo 7 para apresentar e discutir os resultados de experimentos feitos em tempo real para testar o desempenho de cada um dos controladores projetados e implementados. Também se realiza uma comparação do processo de projeto e dos resultados obtidos entre cada uma das metodologias usadas.

Finalmente no capítulo 8 se faz uma conclusão do projeto como um todo e se mencionam sugestões para a continuação de trabalhos futuros.

2 Funcionamento do Simulador de Voo

Neste capítulo serão apresentados de forma resumida os diferentes sistemas e subsistemas que conformam o simulador de voo, e com mais detalhe o sistema do qual o robô paralelo faz parte, uma vez que é o foco deste trabalho. Desde já se esclarece que alguns sistemas ainda não foram implementados e outros não foram implementados totalmente ou ainda estão em desenvolvimento até a publicação deste documento.

O simulador de voo tem como proposito, simular da forma mais fidedigna as sensações que um piloto teria, caso se encontra-se numa aeronave real. Quando se encontra numa situação real o piloto se baseia nos instrumentos de medição da aeronave e visão que tem do seu entorno na cabine, e estes podem ser implementados com o uso de computadores, telas e outros instrumentos.

No entanto outra sensação importante é as acelerações que o corpo do piloto sente ao realizar diversa manobras com a aeronave, neste contexto, atualmente não possível apenas usar um computador para poder simular sensações de aceleração em todas as direções.

Assim se faz necessário um sistema capaz de replicar a sensação de aceleração no corpo humano. Nos simulador de voo se usa o **Sistema de Movimento** para realizar esta tarefa, seu funcionamento será explicado com mais detalhe na seção subsequente.

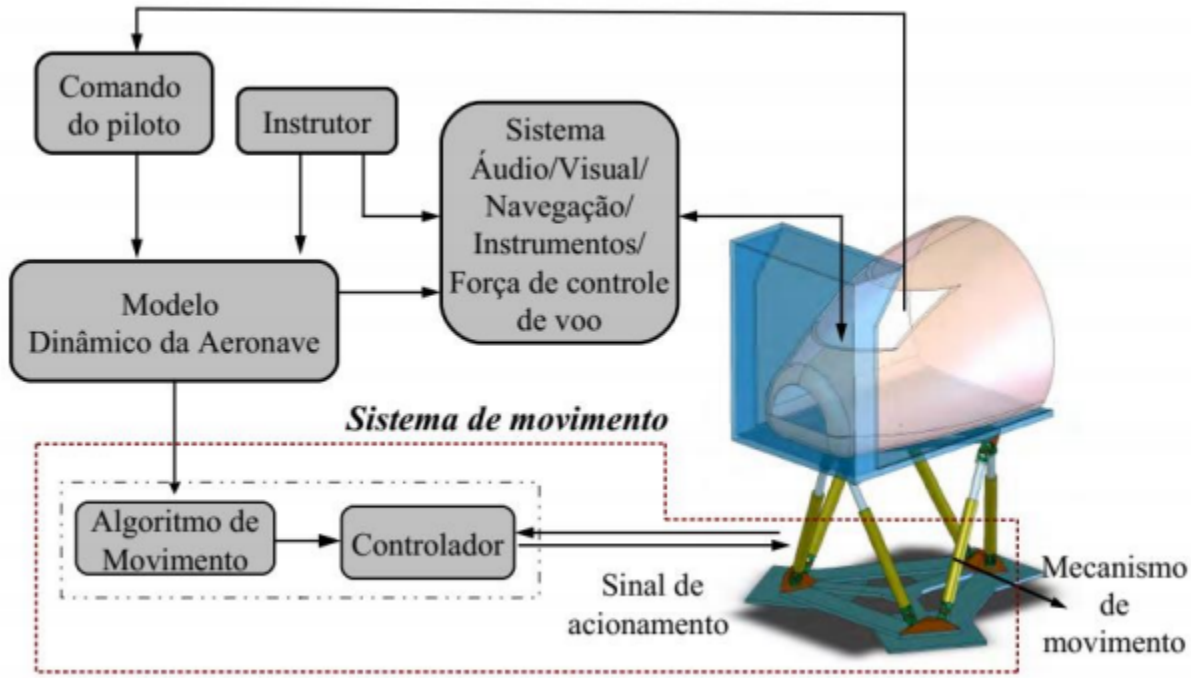


Figura 4 – Estrutura geral do simulador de voo [3]

Seguindo este raciocínio o simulador de voo sendo desenvolvido no GASI, tem a estrutura demostrada na figura 4. O Sistema Audiovisual, Sistema de Navegação, Instrumentos e Força de controle de voo ainda não estão totalmente desenvolvidos, e no momento se usa o *software* de simulação de voo *X-Plane* 10 em uma *workstation*, dedicada unicamente a esta tarefa, para implementar parcialmente estes sistemas.

Como o Sistema de Movimento é o mais significativo para este trabalho ele será exposto com mais detalhes na próxima seção. Os outros sistemas serão apresentados na seção posterior, de forma mais sucinta, apenas para contextualizar seu papel no simulador de voo e suas interações com Sistema de Movimento. Por ultimo se explicara a IMU disponível e seu papel teórico no projeto, embora não seja uma parte ativa do simulador, pois apenas tem papel de análise de desempenho dos diferentes sistemas que o compõem.

2.1 Sistema de Movimento

O Sistema de Movimento é usado para simular as acelerações que o corpo do usuário estaria sujeito numa situação real. Uma forma seria criar estas acelerações, acelerando o próprio corpo do piloto, no entanto seria difícil realizar isto quando se tem acelerações presentes durante um período de tempo relativamente longo, já que o corpo se movimentaria uma distancia igual à integral dupla no tempo desta aceleração.

Uma forma muito conveniente de resolver este problema é usar a sensação de "aceleraçã" causada pela gravidade e orientar o corpo do piloto de forma tal a direcionar a aceleração da gravidade num direção desejada para simular uma aceleração fictícia de longa duração. No entanto ainda é desejável acelerar a massa do piloto para simular acelerações de curta duração.

Assim o Sistema de Movimento implementado no GASI tem um **Mecanismo de Movimento**, onde o corpo do usuário seria acelerado e orientado em relação à aceleração da gravidade; um **Algoritmo de Movimento** que se encarregaria de decidir quando e como o mecanismo deve se movimentar, dependendo do movimento da aeronave simulada (**Modelo Dinâmico da Aeronave** da figura 4); e um **Controlador** que envia o **Sinal de Acionamento** necessário para garantir que o Mecanismo de Movimento siga a referência determinada pelo Algoritmo de Movimento

2.1.1 Algoritmo de Movimento

O Algoritmo de Movimento é uma modelo matemático que, a partir das dinâmica da aeronave simulada pelo Modelo Dinâmico da Aeronave, calcula os movimentos que à plataforma deverá realizar para "enganar" os sentidos do piloto e simular uma sensação de movimento.

O algoritmo usado é denominado de Filtro *Washout*, e transforma os movimentos da aeronave simulado em movimentos que o robô consiga realizar considerando suas limitações físicas. É denominado de "filtro" pois é composto por vários filtros como se observa na figura 5.

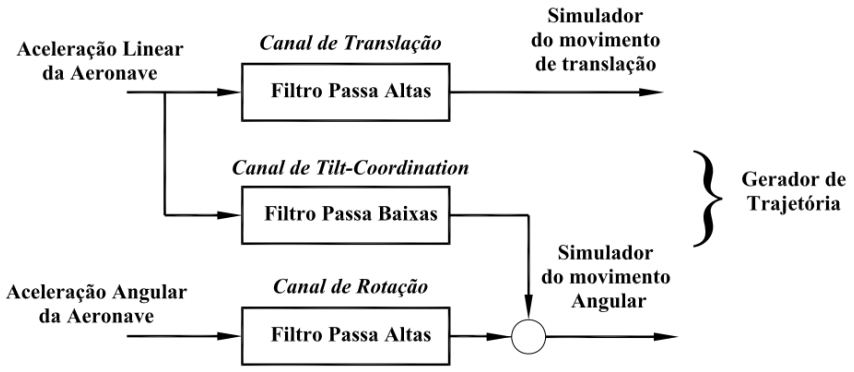


Figura 5 – Funcionamento do filtro Washout do Algoritmo de Movimento [3]

Como explicado anteriormente o algoritmo usa as acelerações da aeronave para determinar o movimento (de translação e angular) da plataforma. Ele é divido em três canais, cada canal tem um filtro diferente. O movimento angular da plataforma é usado para simular componentes de baixa frequência da aceleração linear e angular. Já o deslocamento da plataforma se usa para simular apenas frequências altas da aceleração linear.

O Canal de Translação transforma movimento de traslação rápidos da aeronave, em deslocamentos no robô, respeitando a área de trabalho dele. O Canal de Rotação se encarrega de simular a velocidade angular da aeronave com movimentos de rotação da plataforma de velocidade baixa, apenas perceptíveis pelo corpo.

O Canal *Tilt-Coordination* (Coordenação de Inclinação), se encarrega de transforma acelerações constantes de longa duração em movimentos angulares no mecanismo para usar a gravidade. Estes movimentos devem ser feitos com uma velocidade angular baixa de tal forma que o piloto não perceba a mudança da orientação. Este filtro "coordena", com o canal de rotação, o movimento angular da plataforma.

O sistema de controle não tem nenhuma comunicação com o filtro *washout*, ou seja, o algoritmo em si não tem forma de saber se o Sistema de Movimento como um todo está realizando seu papel de forma satisfatória. Isto reforça à importância do sistema de controle, pois da forma implementada o algoritmo de movimento tem o papel único de gerar referências que a plataforma deve seguir, e é papel do controlador garantir isto. Existem métodos para integrar o algoritmo de movimento com o sistema de controle, mas estes não foram estudados.

O uso do algoritmo de movimento é opcional quando se deseja testar o Sistema de Controle. Por esta razão, não será explicado mais a fundo a lógica por trás dele. Se esclarece que este algoritmo, ainda está sendo implementado por outros alunos, e o autor

não teve participação neste processo. Informações teóricas mais aprofundadas se podem obter na bibliografia [3].

2.1.2 Mecanismo de Movimento

O Mecanismo de Movimento é constituído pelo robô paralelo de 6 GDL, similar à plataforma de Stewart, com topologia 6 U$\underline{P}$UR [3], como pode ser apreciado na figura 6. Isto quer dizer que as juntas existentes em cada uma das "pernas" do robô, começando desde a parte inferior do mecanismo, tem uma junta universal, seguida por uma junta prismática, e um conjunto de juntas universal e de rotação, que em grupo tem um comportamento análogo a uma junta esférica.

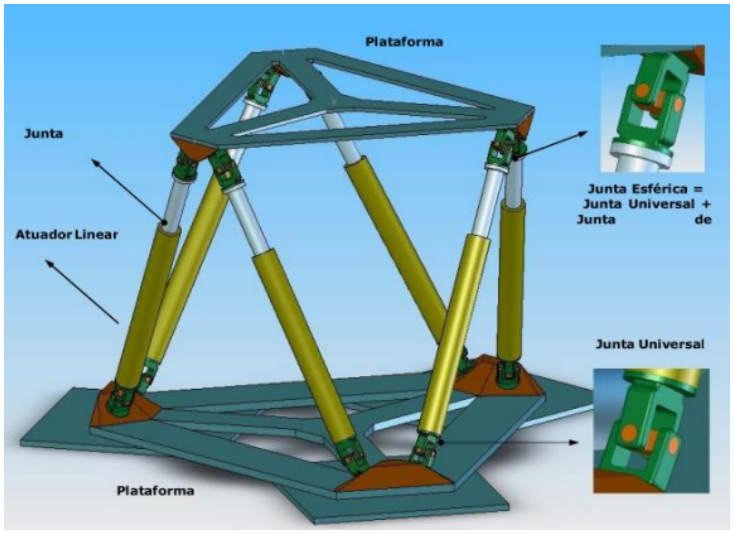


Figura 6 – Topologia 6 U$\underline{P}$UR do robô paralelo [3]

A junta prismática é a junta ativa, ou seja, é nela que se tem atuação. Assim na junta prismática se tem acoplado um motor CC de 12 V [11], que em conjunto de um mecanismo de engrenagens transformam o movimento de rotação do motor em um movimento de translação na junta prismática.

Juntamente com o motor se tem um potenciômetro para medir a traslação que ocorre na junta prismática, este sinal pode ser usado como realimentação para o Controlador. Também é possível usar uma IMU, mas isto será discutido no capítulo seguinte.

Com a atuação de cada motor nas juntas prismáticas das pernas é possível que o robô tenha 6 GDL, para um dado sistema de coordenadas com origem em um ponto da plataforma superior deste, estes são: a liberdade de traslação nos três eixos $\mathbf{x}$, $\mathbf{y}$ e $\mathbf{z}$ de um hipotético sistema de coordenadas e a liberdade de rotacionar em cada um destes eixos para orientar este sistema de coordenadas em relação a um sistema fixo na plataforma inferior.

Usualmente na aviação a orientação deste sistema de coordenadas móvel se usa três variáveis com os nomes de *roll*, *pitch* e *yaw* (também chamado de *heading*), conhecidos como **ângulos de Euler** [4]. Estas variáveis podem ser medidas em radianos ou graus. Esta mesma convenção é usada na plataforma assim os graus de liberdade dela são $[x, y, z, roll, pitch, yaw]$ ou usando uma notação mais curta $[x, y, z, \phi, \theta, \psi]$ [3]. O sistema de coordenadas móvel e de orientação usado na aviação, e no robô, é ilustrado na figura 7.

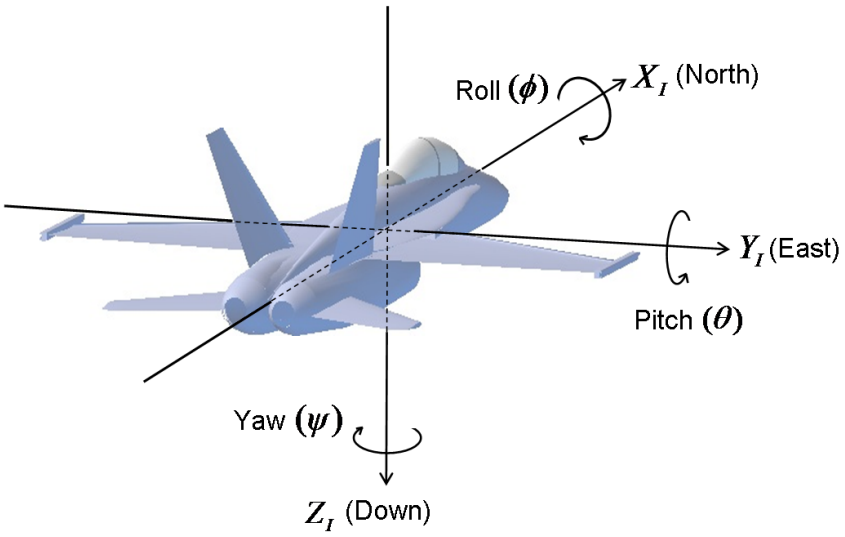


Figura 7 – Sistema de coordenadas móvel e de orientação da aviação [4]

Na figura 8 se observa a plataforma usada como Mecanismo de Movimento para fazer os testes práticos neste trabalho. Se ressalta que este mecanismo já estava construído antes do começo deste projeto e que não se teve participação na construção do mesmo.

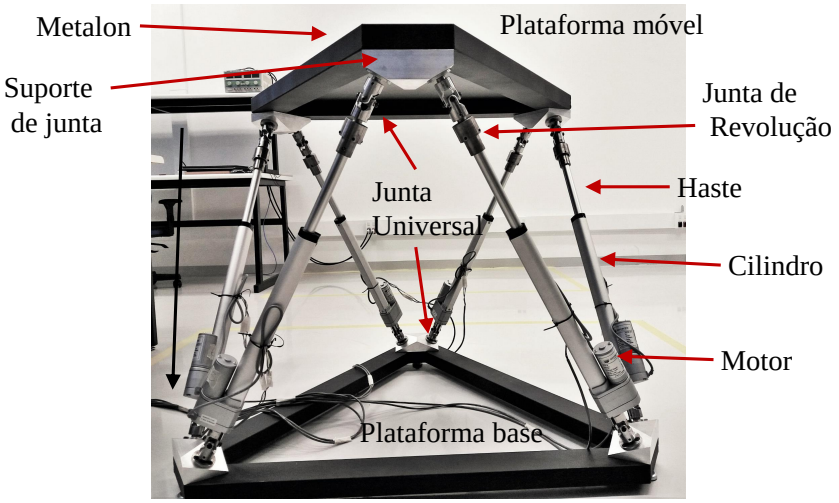


Figura 8 – Robô paralelo de 6 GDL do GASI [3]

2.1.3 Sistema de Controle

O **Sistema de Controle**, no contexto do simulador de voo, tem como tarefa fazer que o Mecanismo de Movimento se translade da forma que o Algoritmo de Movimento indica. Sendo assim se pode abstrair o Algoritmo de Movimento a uma trajetória de orientação e posição que se deseja seguir, ou **Referências** $[x, y, z, \phi, \theta, \psi]$, em conjunto com um **Sinal de Realimentação** da posição e orientação da plataforma se pode obter o **Erro** de seguimento de referência.

Com este erro é possível implementar várias estratégias de controle, estas serão apresentadas com mais detalhes no capítulo 3. O controlador calcularia um **Sinal de Controle** que seria enviado ao Mecanismo de Movimento ou **Planta**, o qual causaria uma dinâmica. Esta dinâmica é medida por um transdutor e se geraria o Sinal de Realimentação, fechando a **Malha de Controle**.

Com estas informações podemos apresentar a organização do controlador, ou de forma mais geral do **Sistema de Controle**. Este sistema possui basicamente três subsistemas, a **Plataforma de Stewart**, a **Interface** e um **Computador**, isto é ilustrado na figura 9.

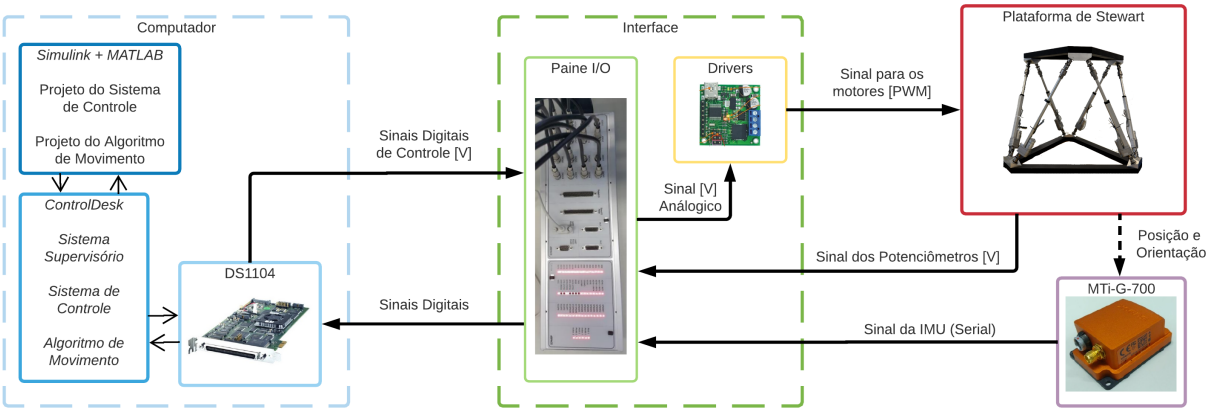


Figura 9 – Organização do Sistema de Controle

2.1.3.1 Plataforma de Stewart

A plataforma contem os atuadores (motores CC [11]) e transdutores (Potenciômetros ou IMU), ela seria a planta da malha de controle, para a qual se manda o sinal de controle e da qual se obtém o sinal de realimentação. Estas informações foram detalhadas na subseção anterior

Devido a limitações de técnicas e questões de segurança do computador, não é viável que ele mesmo envie o sinal com a potencia necessária para fazer os atuadores se movimentarem, assim como receber as medidas feitas pelo potenciômetro, por exemplo. Portanto é necessário ter uma Interface entre a plataforma e o computador.

2.1.3.2 Interface

A tarefa da interface é enviar um sinal com a potencia essencial para os motores dependendo do sinal de controle enviado pelo computador. Assim como isolar o sinal de realimentação dos potenciômetros e transformar ele a uma tensão mais adequada para enviar ao computador.

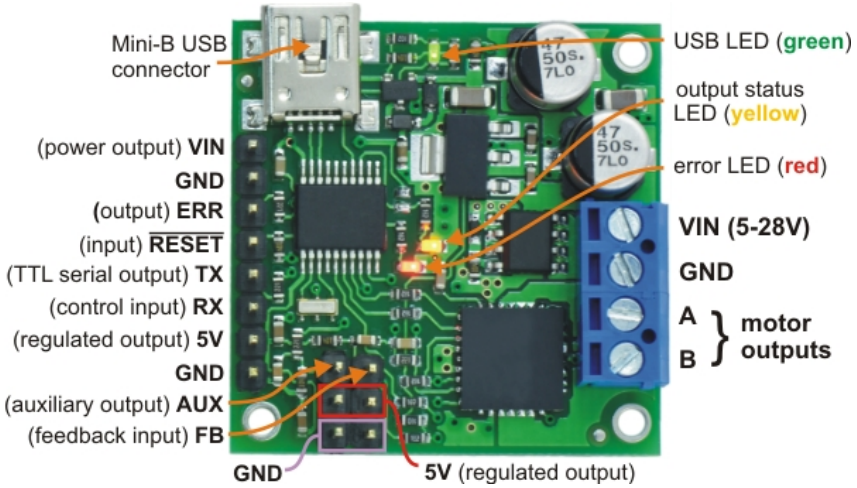


Figura 10 – *Driver* do motor da junta prismática [5]

A interface é realizada usando *drivers* para cada um dos motores, um deles é ilustrado na figura 10. Os *drivers* são alimentados pelas fontes da figura 11, e regulam a velocidade do motor usando um sinal PWM. O *duty cycle* do sinal PWM é regulado usando o sinal de controle de até 5 V que o computador envia.

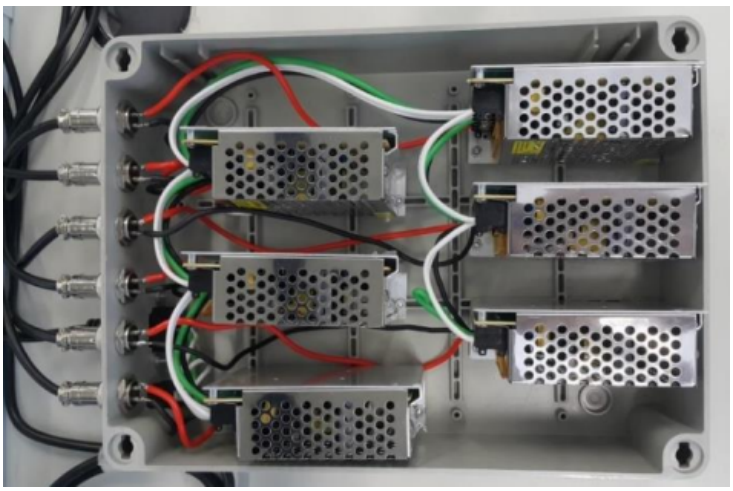


Figura 11 – Caixa de fontes independentes para cada perna

O *driver* é capaz de implementar um controle PID interno de posição ou velocidade do motor, dado que se realimente um sinal adequado para isso a ele [5]. A sintonização do PID pode ser feita, em um computador, individualmente para cada *driver* usando a entrada USB e o *software* disponibilizado pelo fabricante. No entanto está configuração não foi usada durante o desenvolvimento do projeto.

Os potenciômetros são alimentados com a mesma tensão de 12 V, logo a tensão medida vai de 0 a 12 V, contudo está é demasiado alta para enviar ao computador assim o *driver* redimensionada para um sinal analógico com escala de 0 a 5 V, mais compatível com o computador.

2.1.3.3 Computador

O computador contem o algoritmo de controle que seria o controlador da malha. Para simplificar a comunicação entre o Algoritmo de Movimento e o Sistema de Controle, ambos são implementados no mesmo computador e no mesmo ambiente. Ambos são executados por uma placa controladora DS1104 R&D (figura 12), está placa é análogo a um computador dedicado exclusivamente à execução de sistemas de controle e aquisição em tempo real.

Figura 12 – DS1104 R&D *Controller Board* [6]

A placa é instalada dentro de um computador comum, mas agora se tem o beneficio de ter um processador, memoria, etc. que trabalham exclusivamente no processo que o usuário quer. Consequentemente se evita problemas de competição por recursos entre os diferentes processos que existem em um computador usual [6].

Figura 13 – Painel I/O da placa DS1104

A DS1104 tem uma entrada customizada na qual se pode conectar um painel de aquisição e distribuição de sinais (figura 13) [12]. Este painel possui conversores ADC e

DAC, circuitos transmissores e receptores integrados, e é possível adquirir e enviar sinais analógicos, digitais ,PWM , Serial, etc. Em si se poderia considerar a placa controladora como parte do subsistema Computador e o painel parte da Interface.

A placa DS1104 tem *plugins* e bibliotecas para o MATLAB e Simulink, de forma que algoritmos e simulações sejam executados exclusivamente por ela. Também é possível, com o uso destas bibliotecas, acessar as várias saídas e entradas conectadas ao painel I/O, e executar algoritmos em tempo real, através do MATLAB ou Simulink. A vantagem disto é que se pode desenvolver facilmente controladores usando a linguagem de *script* do MATLAB e os modelos desenvolvidos no Simulink, com as quais se tem mais experiencia.

Se usa o *software* de experimentação e desenvolvimento de sistemas de controle eletrônicos, ControlDesk 5.3, também criado pela *dSPACE* [13]. O ControlDesk funciona como um sistema SCADA, propiciando um ambiente onde se pode gravar e mostrar os dados coletados pelo painel I/O, assim como comandar os elementos conectados a este mesmo painel. Os supervisores criados neste *software* também são executados na DS1104

Modelos criados no Simulink podem ser "compilado", isto é, transformados a programas de linguagem C ou C++, e usados no ControlDesk. Desta forma os controladores implementados no Simulink podem ser facilmente portados e usados no ambiente do ControlDesk, que é mais adequado para supervisar o funcionamento destes. Um exemplo de supervisor implementado no ControlDesk pode ser visto na figura 14.

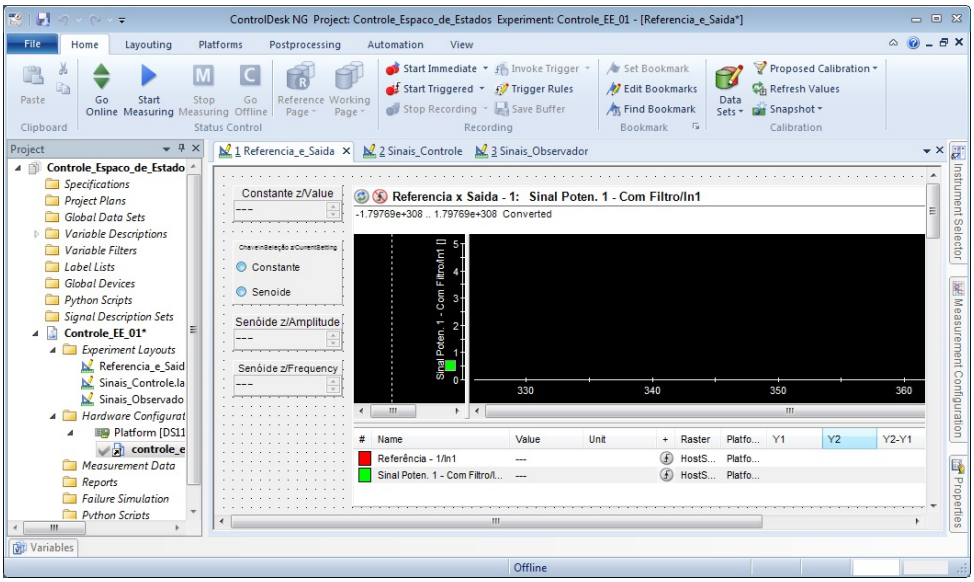


Figura 14 – Exemplo de supervisor implementado no ControlDesk

Se aclara que o tanto o Algoritmo de Movimento como o Sistema de Controle são implementados no Simulink e depois portados para serem usados no ControlDesk.

2.2 Outros Sistemas

O *X-Plane* se encarrega de replicar audiovisualmente o comportamento de uma aeronave, além de simular matematicamente o modelo dinâmico dela. O usuário é capaz de interagir no ambiente virtual do *X-Plane* (Comando do Piloto da figura 4), através de um *joystick*, do teclado e do *mouse*, e consegue perceber o entorno virtual mediante uma tela e a saída de áudio da *workstation*.

Dentro do ambiente criado pelo *X-Plane* se tem a cabine da aeronave com todos os instrumentos que a versão real teria, é possível interatuar com todos estes instrumentos (Sistema de Navegação e Instrumentos da figura 4). Além disto o *software* cria o ambiente onde a aeronave estaria sendo pilotada, isto é, pistas de decolagem, os diferentes elementos topográficos presentes no mundo real (montanhas, rios, oceanos, etc.), as condições climatológicas, a horário local (dia, noite, etc.). Um exemplo disto se pode ver na figura 15. Sendo assim o *X-Plane*, garante que se tenha condições nas quais uma aeronave autêntica estaria exposta, tornando a simulação mais real.

Figura 15 – Exemplo de vista da cabine do piloto no *X-plane* 10

O sistema de forças de controle de voo (figura 4), simularia as forças que o piloto sente ao usar os controles da aeronave, similar a força que uma pessoa sente no braços para manter o volante de um carro ao fazer uma curva. No momento estes sistemas não se encontram acoplados na parte superior do Mecanismo de Movimento, como é feito no caso de outros simuladores de voo, devido a limitações técnicas.

Como o *X-Plane* simula matematicamente a dinâmica da aeronave, é possível ter acesso as variáveis que regem o movimento da aeronave como, aceleração linear, aceleração angular, orientação e posição. Usando estas informações é possível emular as sensações de aceleração do corpo do usuário usando o sistema de movimento.

Sendo assim estas informações são enviadas ao Sistema de Movimento através de comunicação serial usando o padrão RS-232. O Sistema de Movimento recebe estes dados pela porta serial do painel I/O.

2.3 A Unidade de Medição Inercial (IMU)

Se usa uma IMU para fazer medições da posição e orientação (ângulos de Euler), além da velocidade linear e angular do Mecanismo de Movimento. Isto é necessário para fazer uma análise do desempenho dos subsistemas do Sistema de Movimento, como o Algoritmo de Movimento e o Sistema de Controle.

A unidade que se tem a disposição é o MTi-G-700, fabricado pela *XSens*. Usando-se a terminologia correta, ele é um sistema de navegação inercial (INS) integrado com um sistema de posicionamento global (GPS) [14]. Para efeitos deste trabalho, como algumas características dele não são empregados neste projeto, pode-se considerá-lo como uma IMU. Na figura 16 se pode reparar a IMU usada neste projeto.

Figura 16 – MTi-G-700 usado no projeto

Considera-se a IMU como um elemento passivo, pois não tem um papel crítico ao funcionamento do simulador como um todo, ou seja, o simulador funciona normalmente sem este elemento. É concebível usá-la para realizar a realimentação no Sistema de Controle, mas, no momento, não é viável devido a que o MTi-G-700 se comunica usando a serial e apenas se pode usar uma das portas seriais do painel I/O (figura 13), quer dizer, não é possível utilizar as duas ao mesmo tempo, porque são de padrões de comunicação serial diferentes e só se pode usar um padrão no mesmo programa.

Informações mais detalhadas do funcionamento do MTi-G-700 podem ser vistas nas referências [15].

2.4 Resumo do Capítulo

O simulador desenvolvido no GASI possui vários sistemas, como se pode observar na figura 4. Entre eles, o projeto descrito neste documento pertence ao **Sistema de Movimento**, porque é nele que se encontra a **Malha de Controle**. Consequentemente este sistema e seus componentes foram amplamente descritos na Seção 2.1.

Dentro deste sistema se tem três subsistemas, o **Algoritmo de Movimento**, o **Sistema de Controle** e o **Mecanismo de Movimento**. O subsistemas mais críticos para o desenvolvimento do trabalho são o Sistema de Controle e o Mecanismo de Movimento, já que eles compõem a maior parte da Malha de Controle, como a Planta, o Sistema de Controle e o Sensor. O Algoritmo de Movimento, apesar de também fazer parte da malha, não é tão importante, pois pode ser abstraído simplesmente como a Referência da malha.

Para melhor entender o funcionamento do simulador como um todo, se expôs de maneira sucinta as atividades dos **outros sistemas** que o integram na Seção 2.2. Estes sistemas ainda não estão prontos. No momento se usa o ***X-plane* 10**, para simular o **Modelo Dinâmico da Aeronave**, que se comunica diretamente com o Algoritmo de Movimento.

Para projetar o Sistema de Controle e o Algoritmo de Controle se usa o **Simulink**, em conjunto com a linguagem de *script* do **MATLAB**, em um modelo que é compilado para ser usado no ControlDesk. O **ControlDesk** é usado como um sistema supervisor, onde se controla e mostra os diferentes sinais presentes no Sistema de Controle em tempo real. Estes sinais são transmitidos ao **painel I/O** (figura 13), que está conectado diretamente à **placa controladora DS1104**.

A DS1104 pode ser considerado com um minicomputador, onde os algoritmos do Sistema de Movimento são executados, desta forma se evita problemas de falta de recursos, pois ela estaria executando unicamente as operações imprescindíveis para o funcionamento deste.

Um elemento adicional que seria usado no projeto é o MTi-G-700, que para efeitos deste trabalho é utilizado como uma IMU. Este elemento é descrito na seção 2.3, e é usado para realizar medidas da posição e orientação **apenas** para testar o desempenho do sistema de movimento individualmente. Isto se deve, a limitação física do Painel I/O, que pode usar apenas um padrão de comunicação serial ao mesmo tempo.

Uma particularidade do Sistema de Movimento é que ele não necessita estar funcionando em conjunto com os outros sistemas para testar seu desempenho, pois se pode gravar dados do Modelo Dinâmico da Aeronave em uma ocasião e usar eles para fazer testes mais adiante. O mesmo se aplica para a Malha de Controle, isto é, se pode gravar as referências geradas pelo Algoritmo de Movimento e usá-las depois para realizar testes homogêneos.

3 Métodos para Projetar o Sistema de Controle

No capítulo a seguir se explicitarão as métodos teóricos que serão usados para projetar o Controlador do Sistema de Movimento do robô.

Como foi explicado na capítulo anterior, se pode considerar a função do controlador como um problema clássico, na robótica, de seguimento de trajetória. Portanto se pode usar os métodos conhecidos na literatura para resolver este problema [7]. Estes serão descritos detalhadamente neste capítulo. Serão denominados de **Estruturas de Controle** os métodos que determinam a estrutura que a malha de controle terá, porém deixando indeterminado o tipo de algoritmo de controle a ser usado.

A **Metodologia de Controle** se refere ao algoritmo de controle, isto é, à forma como ocorre o seguimento de referência. Neste caso se pode usar a bibliografia sobre teoria controle [16] [8]. Estes métodos se referem a estrategia com a qual será projetado o controlador em si, como por exemplo, controlador PID, controle por realimentação de estados, LQR, controladores não lineares, controladores robustos, controle por lógica fuzzy, controle por modos deslizantes, etc. [17] [18] [19] [20]

As metodologias nem sempre dependem da estrutura de controle usada, contudo algumas metologias requerem que determinados sinais sejam realimentados e/ou que se tenham modelos mais complexos, como por exemplo no caso de usar controladores não lineares.

Dentro das estruturas de controle se usam as teorias de **Cinemática de Robôs**, logo será necessário apresentar a teoria, especifica para o caso da plataforma de Stewart, para resolver a o problema de cinemática direta e inversa.

Primeiramente serão definidos com mais detalhes o problema e quais as características requeridas do controlador, para que o sistema de movimento possa funcionar no contexto do simulador de voo. Posteriormente se explicará a cinemática do robô para compreender melhor as estruturas de controle, que serão especificadas em sequencia. Finalmente se realizara uma descrição breve das técnicas para projetar os controladores com as quais se pretende solucionar o problema.

3.1 Definição do Problema

O objetivo do Sistema de Movimento é simular as sensações físicas de movimento no corpo do piloto, ou seja, a sensação de aceleração e orientação que se tem em um voo

real.

Com o Modelo Dinâmico da Aeronave, que faz uma simulação matemática de uma aeronave real, se obtém as informações necessárias para que o Algoritmo de Movimento determine o comportamento do Mecanismo de Movimento.

Isto é conquistado com uso do Controlador, que, dadas as diretrizes transmitidas pelo Algoritmo de Controle (Referência) e os sinais de realimentação do estado do Mecanismo de Movimento, implementa um algoritmo de controle que garantirá o movimento adequado deste.

O mecanismo de movimento de simuladores de voo usualmente trabalham em frequências de até 1 ou 2 Hz para simular mudanças da aeronave causados pelo piloto. Enquanto que distúrbios, como turbulências causam movimentos de até 10 Hz [3]. Este comportamento é determinado pelo Algoritmo de Movimento, ou seja, as referências que mecanismo de movimento dever seguir sempre estarão nessas faixas.

Atraso entre o movimento determinado pelo modelo dinâmico da aeronave e movimento real do simulador devem ser no máximo de 1 a 2 s para garantir que estes não serão percebidos pelo piloto e assim manter a qualidade da simulação [3].

Os movimentos do simulador devem ser suaves, de forma a evitar dar uma sensação falsa do movimento da aeronave ao usuário. Isto quer dizer que não se pode tolerar sobre-sinais quando se faz o seguimento da trajetória [3].

Finalmente o sinal de controle usado dever respeitar os limites físicos dos atuadores a fim de evitar problemas de sobrecorrente e desgaste dos motores.

3.2 Cinemática do Robô

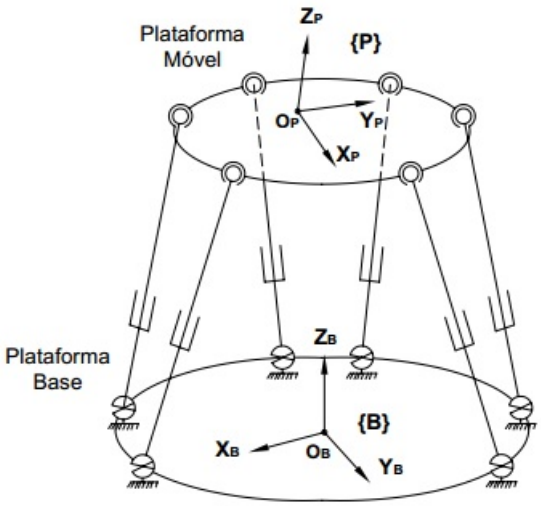


Figura 17 – Sistema de coordenadas definido para a plataforma de Stewart [3]

Essencialmente um manipulador robótico é uma cadeia de corpos rígidos (*links*) conectados por juntas, por definição um destes corpos sempre deve estar restringido a um ponto fixo, este é denominado de **base**. Se escolhe algum corpo livre do manipulador para ser o "atuador terminal" (*end-effector*).

As cadeias cinemáticas são definidas como abertas quando para existe apenas uma sequencia de *links* que conectam quaisquer dois pontos da cadeia. Está definição é transportada para a robótica, onde robôs são denominados de seriais quando possuem esta propriedade. Já robôs paralelos são aqueles que possuem mais de uma sequencia que conecta quaisquer dois pontos da sua cadeia cinemática. Sendo assim, a plataforma de Stewart é considerada como um robô paralelo.

Dois sistemas de coordenadas importantes são definidos no robô, uma **sistema referência fixo** $\{\mathbf{B}\}$, e geralmente se encontra em algum ponto da base. O outro $\{\mathbf{P}\}$, que usualmente se localiza no *end-effector*, que descreve o movimento desse corpo, denominado de **sistema móvel**. Para a plataforma de Stewart estes sistemas estão definidos pela figura 17.

Se define uma matriz $\mathfrak{R}$, chamada de matriz de rotação, que se usa para transformar vetores em um sistemas de coordenas para outro que se encontra rotacionado em relação ao primeiro.

3.2.1 Cinemática Inversa

Define-se como cinemática inversa ao método usado para determinar o comprimento ou rotação que os atuadores devem ter para que o sistema de coordenadas móvel tenha uma determinada posição e orientação. Se define estas coordenadas de posição e/ou orientação como o **espaço cartesiano**, enquanto que os comprimentos e/ou rotações das juntas como o **espaço das juntas**.

Em essência a cinemática inversa converte um determinado vetor no espaço de coordenadas para o espaço das juntas, ou seja, quais movimentos as juntas ativas devem realizar para fazer que o robô assuma a posição e orientação determinada por esse vetor $[x, y, z, \phi, \theta, \psi]$.

Na plataforma de Stewart se pode usar os seguintes vetores, definidos pela figura 18 para descrever a cinemática inversa. Onde:

- **t** é o vetor, no sistema de coordenadas fixo, que descreve a posição da origem do sistema de coordenadas móvel da plataforma superior;
- **p** é o vetor **constante**, no sistema de coordenadas móvel, que descreve a posição da junta que une a perna à plataforma superior ;

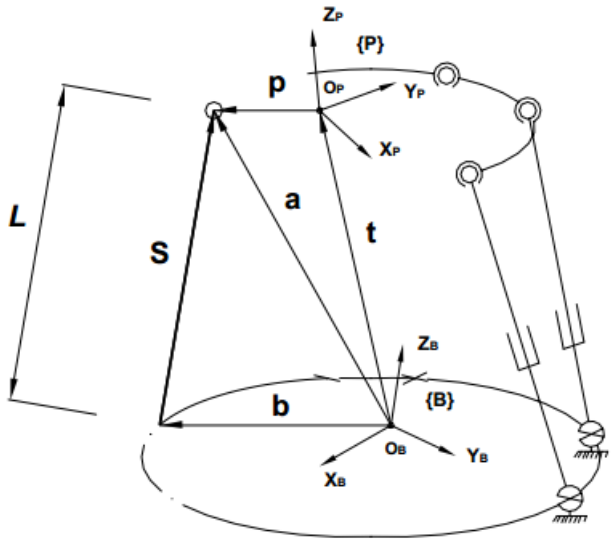


Figura 18 – Diagrama vetorial de uma perna da plataforma de Stewart [3]

- **S** é o vetor, no sistema de coordenadas local da perna, que descreve a posição da junta que une a perna à plataforma superior;
- **b** é o vetor **constante**, no sistema de coordenadas fixo, que descreve a posição da origem do sistema de coordenadas local da perna;
- **a** é o vetor, no sistema de coordenadas fixo, que descreve a posição da junta que une a perna à plataforma superior, igual a $\mathfrak{R}\,\mathbf{p} + \mathbf{t}$;
- L é a variável escalar que determina o comprimento da perna.

Usando estas informações se pode concluir que o vetor **S** pode ser calculado usando uma soma vetorial descrita pela seguinte equação

$$\mathbf{S} = \mathbf{a} - \mathbf{b} = \mathfrak{R}\,\mathbf{p} + \mathbf{t} - \mathbf{b}. \tag{3.1}$$

Assim a norma deste vetor, que define o vetor **t** no espaço da junta local, determina o valor de

$$L = \|\mathbf{S}\|\,. \tag{3.2}$$

Generalizando as equações 3.2 e 3.1 para todas as pernas, se pode definir a cinemática inversa da plataforma de Stewart como as seguintes equações

$$\begin{aligned} \mathbf{S}_i &= \mathfrak{R}\,\mathbf{p}_i + \mathbf{t} - \mathbf{b}_i, \\ L_i &= \|\mathbf{S}_i\| \ \text{ para } \ i = 1, 2, ..., 6. \end{aligned} \tag{3.3}$$

3.2.2 Cinemática Direta

Oposto à cinemática inversa, a cinemática direta transforma um vetor de valores no espaço das juntas para um vetor de coordenadas no espaço cartesiano. Diferente da anterior, a aplicação da cinemática direta para a plataforma de Stewart, não tem uma solução algebraica, mas se pode resolver usando algoritmos recursivos [3].

Isto é devido a sua classificação como um robô paralelo, já que para este tipo à resolução da cinemática inversa é um problema simples, mas a cinemática direta tem uma solução complicada. Para robôs seriais o contrario é verdade [7].

Analisando a equação 3.3 se pode provar que ao escrever o vetor $\mathbf{t}$ em relação aos outros termos, se cria um sistema indeterminado formado por equações não lineares, que possuem múltiplas soluções. Isto dificulta à solução da cinemática inversa e ao mesmo tempo dificulta implementar algoritmos de controle em tempo real. Soluções numéricas foram revisadas e se encontram na bibliografia [3], mas não serão mais discutidas, pois não serão usadas.

3.3 Estruturas de Controle

Como se mencionou anteriormente o termo **Estrutura de Controle**, se usa no sentido a descrever a estrutura que a malha de controle terá, sem necessariamente determinar o tipo de algoritmo de controle a ser usado nesta malha, para um dado problema de seguimento de referência (em coordenadas cartesianas).

Na literatura existem diversos tipos de estruturas de controle, desenvolvidas especialmente para usar em sistemas robóticos. A maioria das estruturas podem ser usadas tanto em robôs seriais como em paralelos, não obstante algumas estruturas são mais fáceis de ser implementadas em certos casos [7].

As estruturas de controle que foram consideradas são três, **Controle no Espaço das Juntas**, **Controle no Espaço Cartesiano**; e **Controle baseado na Dinâmica Inversa**. A escolha destas estruturas se deve a que estas foram as analisadas pelo Prof. Mauricio no seu doutorado e portanto se tinha mais informações delas [3].

3.3.1 Classificação

As estruturas podem ser classificadas em dois tipos dependendo da forma como consideram o sistema robótico. Em estruturas **descentralizadas**, o controle se realiza considerando o manipulador como um conjunto de n sistemas independentes (n juntas ativas). O controlador usado neste caso seria do tipo SISO, onde acoplamentos entre os sistemas são considerados como perturbações [7].

Nas estruturas **centralizadas**, são baseados no conhecimento, parcial ou total, da dinâmica do mecanismo. Em essência todos os sistemas robóticos são sistemas MIMO não lineares com n entradas (n juntas ativas) e m saídas. Tendo isto em conta este tipo de estrutura considera os acoplamentos como parte do sistema e portanto os compensa mais rapidamente [7].

Existe ainda a classificação segundo espaço com no qual se define o erro de seguimento, no qual existem duas categorias.

Estruturas no **espaço cartesiano**, quando o erro é calculado comparando as coordenadas de referência com da medida das coordenadas cartesianas do robô.

Estruturas no **espaço das juntas** trabalham com erro calculado a partir da comparação entre uma referência no espaço das juntas. O espaço das juntas no caso da plataforma de Stewart são os comprimentos das seis juntas prismáticas.

3.3.2 Controle no Espaço das Juntas

A estrutura de controle no espaço das juntas é um esquema geral de controle, que pode ser do tipo centralizado ou descentralizado, dependendo do tipo de algoritmo de controle implementado (SISO ou MIMO). Esta estrutura pode ser observada na figura 19.

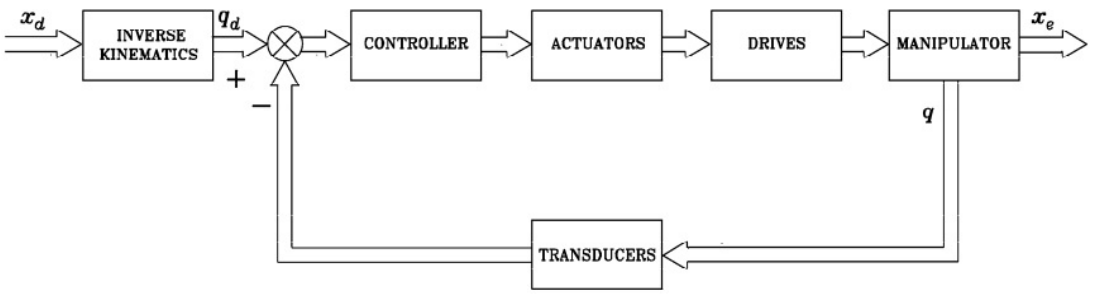


Figura 19 – Estrutura geral do controle no espaço das juntas [7]

O esquema se divide em dois subproblemas. O calculo da cinemática inversa (*Inverse Kinematics*) é usado para transformar as coordenadas cartesianas (espaço cartesiano) desejadas $\boldsymbol{x_d}$, para uma referência no espaço das juntas $\boldsymbol{q_d}$.

Em conjunto com a medida do transdutor das juntas $\boldsymbol{q}$, se pode calcular o erro de seguimento e usar ele no controlador. O *design* deste controlador é o segundo problema desta estrutura.

A vantagem deste esquema é o uso de referências no espaço das juntas, que pode ser comparado com o valor lido pelo transdutor integrado na junta, ou seja, não é necessário medir diretamente as coordenadas cartesianas do mecanismo, o que exigiria uma transdutor sofisticado, como uma IMU.

A desvantagem deste tipo é que incertezas do modelo tem bastante influencia no desempenho do controlador, pois não se controla diretamente $\boldsymbol{x_e}$. Uma outra desvantagem é o calculo da cinemática inversa que em certos mecanismos é um problema computacional complexo, como no caso dos robôs seriais, mas no caso dos paralelos é mais simples, como foi discutido na seção 3.2.

3.3.3 Controle no Espaço Cartesiano

A estrutura de controle no espaço cartesiano se pode ver na figura 20. Neste caso se realimenta as coordenadas cartesianas do robô $\boldsymbol{x_e}$ e se compara diretamente com as desejadas $\boldsymbol{x_d}$.

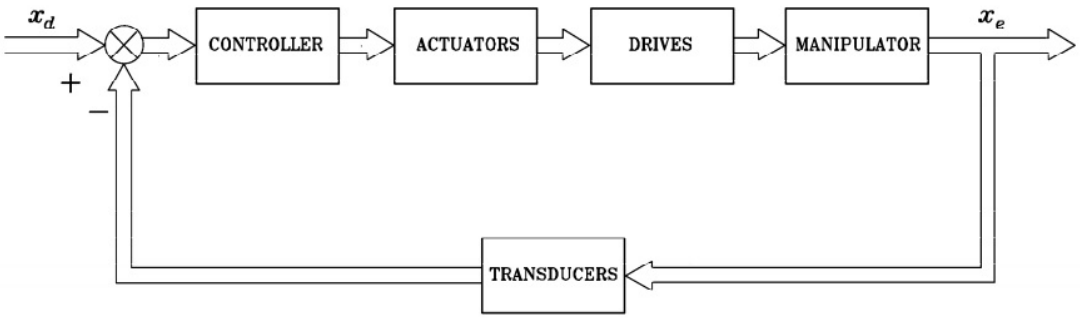


Figura 20 – Estrutura geral do controle no espaço cartesiano [7]

Esta estrutura de controle embora tenha, aparentemente, um esquema mais simples, geralmente tem uma complexidade algorítmica do controlador maior, pois a cinemática inversa já estará embutida dentro do próprio controlador, já que a atuação (sinal de controle) sempre se dá no espaço das juntas. A vantagem desta estrutura é o melhor desempenho da malha inteira quando se observar a saída, pois o controle já leva em conta o acoplamento entra as juntas.

Uma desvantagem é a difícil medição das coordenadas cartesianas do sistema. Este problema pode ser contornado com o uso da cinemática direta e as medidas do transdutor no espaço das juntas.

3.3.4 Controle Baseado na Dinâmica Inversa

Esta estrutura se baseia no modelo dinâmico do robô. Assumindo que este modelo é dado pelo sistema multivariável de equações dado por

$$\boldsymbol{B}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{u}. \tag{3.4}$$

Onde $\boldsymbol{B}(\boldsymbol{q})$ e $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ são funções matriciais. Assumindo que a entrada $\boldsymbol{u}$ tem o seguinte comportamento

$$\boldsymbol{u} = \boldsymbol{B}(\boldsymbol{q})\boldsymbol{y} + \boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}}), \tag{3.5}$$

e substituindo na equação 3.4, onde $\boldsymbol{y}$ é uma nova entrada a ser definida, se obtém

$$\ddot{\boldsymbol{q}} = \begin{bmatrix} \ddot{q}_1 \\ \vdots \\ \ddot{q}_n \end{bmatrix} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} = \boldsymbol{y}. \tag{3.6}$$

Assim se transforma o problema não linear e acoplado, em um linear desacoplado, onde a integral dupla de cada y_i afeta apenas um atuador q_i. Por causa da definição do controle dado pela equação 3.5, se dá o nome de controle baseado na dinâmica inversa.

Como se quer que se siga uma referência $\boldsymbol{q_d}$, então o erro de seguimento é dado por

$$\tilde{\boldsymbol{q}} = \boldsymbol{q_d} - \boldsymbol{q}, \tag{3.7}$$

e escolhemos $\boldsymbol{y}$ como sendo

$$\boldsymbol{y} = \ddot{\boldsymbol{q}}_{\boldsymbol{d}} + \boldsymbol{K}_D\dot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_P\tilde{\boldsymbol{q}}, \tag{3.8}$$

onde são matrizes constantes. Usando estas equações e substituindo em 3.6, obtemos

$$\ddot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_D\dot{\tilde{\boldsymbol{q}}} + \boldsymbol{K}_P\tilde{\boldsymbol{q}} = 0. \tag{3.9}$$

A equação 3.9, é uma EDO homogênea que define o comportamento do erro ao longo do tempo. Estipulando que $\boldsymbol{K}_D$ e $\boldsymbol{K}_P$ sejam digonais positivas, pode-se afirmar que o erro $\boldsymbol{q_d}$ converge a zero ao longo do tempo com velocidade definida pelos elementos dessas matrizes.

A figura 21 apresenta esta estrutura na forma de diagrama de blocos. No diagrama os blocos que contem $\boldsymbol{B}(\boldsymbol{q})$ e $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ são os que linearizam e desacoplam o sistema, enquanto que as matrizes $\boldsymbol{K}_D$ e $\boldsymbol{K}_P$ estabilizam o sistema.

Nota-se que esta estrutura é centralizada e no espaço das juntas, contudo é necessário ter as derivadas das saída e das referências, como se observa na figura 21, o que não visto na estrutura de controle no espaço das juntas vista na figura 19. A grande desvantagem desta estrutura é a complexidade que se requere para determinar as funções $\boldsymbol{B}(\boldsymbol{q})$ e $\boldsymbol{n}(\boldsymbol{q}, \dot{\boldsymbol{q}})$.

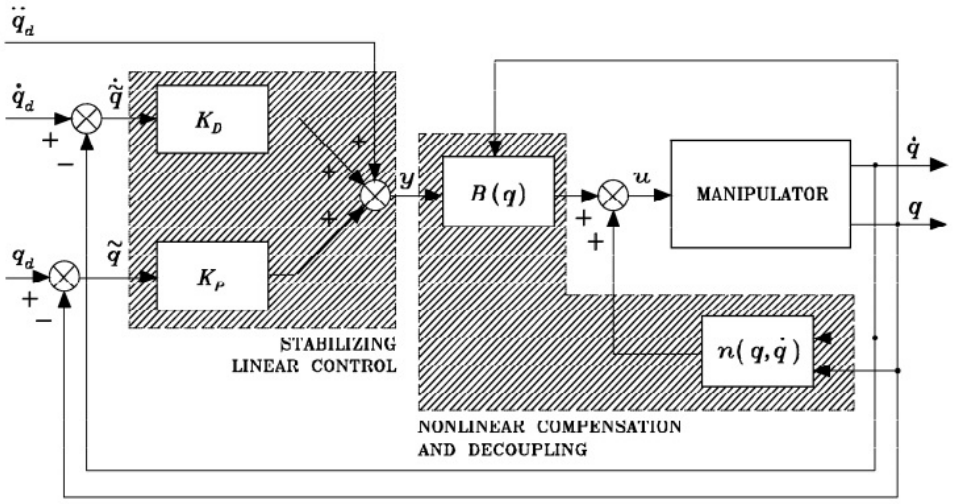


Figura 21 – Estrutura geral do controle baseado na dinâmica inversa [7]

3.4 Estratégias de Controle

Como mencionado anteriormente, **estrategia de controle** se refere ao método usado para garantir o seguimento de referência, quando se usa uma determinada estrutura de controle. Uma analogia que se pode fazer é, a estrutura de controle é o diagrama de blocos a ser usado para realizar o controle, enquanto que a estrategia é a metologia pela qual se projeta o controlador desta malha.

Algumas estruturas de controle permitem o uso de uma variedade de estrategias de controle, enquanto o outras são menos flexíveis, como no caso da estrutura de controle baseado na dinâmica inversa.

As estrategias consideradas para desenvolver o controlado serão apresentadas, de forma breve, a seguir. Elas foram agrupadas em dois grupos, **controle clássico** e **controle moderno**, dependendo da forma como se trabalha matematicamente os modelos dos sistemas.

3.4.1 Controle Clássico

No controle clássico, as equações diferencias que descreve o comportamento de um sistema no domínio do tempo, são transformadas para o domínio da frequência usando a transformada de Laplace. Isto é realizado devido às propriedades benéficas que se apresentam neste domínio, que facilitam o projeto de controle para sistemas lineares.

A teoria clássica de controle tem várias formas de projeto de controladores, logo seria inconveniente revisar todos para o projeto, assim se usam as técnicas mais conhecidas.

3.4.1.1 Controle Proporcional Integral Derivativo (PID)

O PID é uma dos controladores mais antigos, contudo ainda continua sendo o mais usado na industria, em razão da sua facilidade de implementar e por seu bom desempenho na maioria dos problemas de controle.

Sistemas de malha fechada com controle PID, mostrado na figura 22, tem três termos, o termo proporcional, integrativo e derivativo.

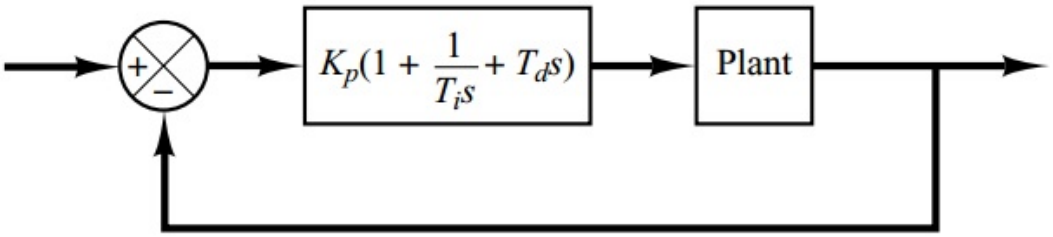


Figura 22 – Sistema de malha fechada com controlador PID [8]

Como se pode observar na seguinte equação

$$C(s) = K_p(1 + \frac{1}{T_p s} + T_d s) = K_p + \frac{K_i}{s} + K_d s = \frac{K_d s^2 + K_p s + K_i}{s}, \tag{3.10}$$

o PID $C(s)$ ideal não é realizável fisicamente devido a que possui mais zeros do que polos, portanto se adiciona um polo ao termo derivativo, tornando-o um filtro derivativo. Assim a função de transferência do controlador passa a ser

$$C(s) = K_p + \frac{K_i}{s} + K_d \frac{s}{T_f s + 1} = \frac{(K_d + K_p T_f)s^2 + (K_p + K_i T_f)s + K_i}{T_f s^2 + s}, \tag{3.11}$$

onde T_f é a constante de tempo do filtro derivativo, K_p é o ganho proporcional, K_i é o ganho integrativo, e K_d é o ganho derivativo.

O PID garante seguimento de referência devido a seu termo integrativo, pois ele faz com que a dinâmica do erro de seguimento se torne zero ao longo do tempo.

Existem diversos métodos para sintonizar os parâmetros do PID mas eles não serão discutidos aqui, mais informações na bibliografia [8] [16].

3.4.1.2 Método do Lugar das Raízes

Este método visa analisar como as raízes, do sistema em malha aberta ($G(s) = N_G(s)/D_G(s)$), se modifica ao adicionar o compensador $C(s) = N_C(s)/D_C(s)$ em malha fechada com a planta, como se vê na figura 23.

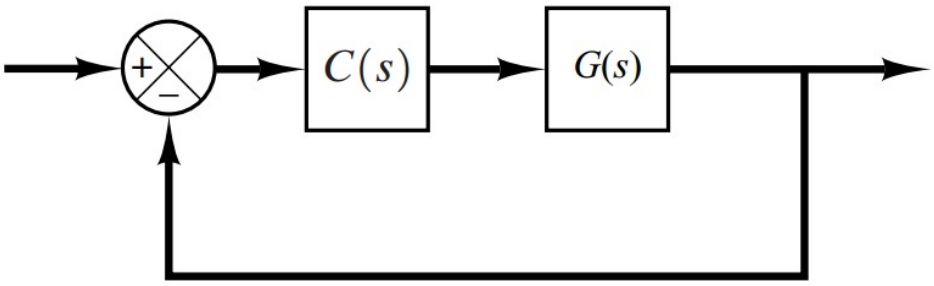


Figura 23 – Sistema de controle em malha fechada [8]

Para isso, se cria o diagrama do lugar das raízes. As regras para criar os diagramas estão disponíveis na bibliografia [8] [16] , e por tal não serão discutidas neste documento.

Por exemplo, a figura 24 mostra como a posição de um zero real no compensador muda o lugar das raízes de um sistema com dois polos imaginários e um real, o segundo polo real é do compensador.

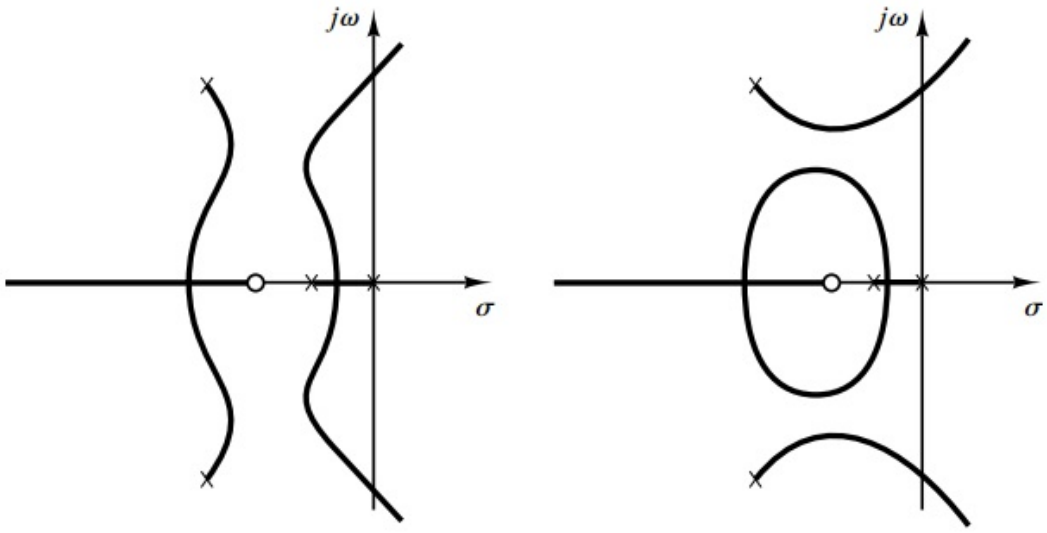


Figura 24 – Lugar das raízes de um sistema com 3 polos, para 2 zeros diferentes [8]

Os parâmetros do compensador são achados usando regras conhecidas do diagrama de polos e zeros, de tal forma que a localização final dos polos de malha fechada cumpra algum requisito de desempenho que se requeira do sistema.

3.4.2 Controle Moderno

O controle moderno, surgiu como uma forma de resolver problemas de controle mais complexos que as técnicas clássicas não conseguiam, ou também para melhorar e otimizar processos que demandam um melhor desempenho.

As EDOs que descrevem o comportamento de uma certa planta já se encontram

no domínio do tempo, porém devido a que muitos sistemas tem EDOs de grau elevado, o que torna sua analise complicada. A forma utilizada para contornar esta dificuldade é a transformação para variáveis de estado, onde uma EDO de grau n é divida em n EDOs de 1º grau.

Isto torna, particularmente, a analise de sistemas MIMO muito fácil, onde se pode representar de forma compacta sistemas de EDOs grandes usando matrizes. No entanto serão só apresentados exemplos para sistema SISO e invariantes no tempo, pois apenas se usaram este tipo no projeto, definidos com a seguinte notação

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\,\mathbf{x}(t) + \mathbf{B}\,u(t), \\ y(t) &= \mathbf{C}\,\mathbf{x}(t). \end{aligned} \tag{3.12}$$

Onde $\mathbf{x}(t)$ é vetor de estados com n estados; $y(t)$ é a função escalar da saída do sistema; $u(t)$ é a entrada controlada definida por uma função escalar; $\mathbf{A}$ é uma matriz constante de dimensão $n \times n$; $\mathbf{B}$ é uma matriz constante de dimensão $n \times 1$; $\mathbf{C}$ é uma matriz constante de dimensão $1 \times n$.

No espaço de estados a maioria dos projetos de controle linear se focam em estabilizar alguma dinâmica do tipo,

$$\begin{aligned} \dot{\varepsilon}(t) &= (\mathbf{A} - \mathbf{B}\,\mathbf{K})\,\varepsilon(t) = 0, \\ \varepsilon(t) &= e^{(\mathbf{A}-\mathbf{B}\,\mathbf{K})\,t}\varepsilon(0). \end{aligned} \tag{3.13}$$

Onde $\mathbf{K}$ é uma matriz constante que o projetista tem liberdade de escolher. O que muda de um método para outro é a forma como o projetista tem a habilidade de manipular esta matriz para que a dinâmica seja estabilizada com um comportamento ideal, como por exemplo, a estabilização da dinâmica do erro de seguimento de referência.

Assim como no caso do controle clássico, serão apresentadas duas metodologias para projeto de controladores, mais métodos podem ser revisados na bibliografia [8] [16]. No entanto, é adequado ressaltar o caso especial de **controle de sistemas integrativos**, pois esta não é teoria muito comum quando se fala de controle no espaço de estados.

3.4.2.1 Controle de Sistemas Integrativos

Dado um sistema SISO, invariante no tempo e que se sabe que tem uma dinâmica integrativa, representado pela equação 3.12. Assumindo que se tenha uma representação apropriada do sistema no espaço de estados, onde a saída $y(t)$ seja igual ao estado $x_1(t)$, e que se trata de um sistema integrativo.

Usando como sinal de controle

$$u(t) = \begin{bmatrix} 0 & k_2 & \cdots & k_{n-1} & k_n \end{bmatrix} \mathbf{x} + k_1(r - x_1) = \mathbf{K}\mathbf{x}(t) + k_1 r(t), \tag{3.14}$$

se pode provar que a dinâmica dos estados é estável, e que a saída tende assintoticamente à um dado sinal de **referência do tipo degrau**, se e somente se os **autovalores da matriz** $(\mathbf{A} - \mathbf{B}\,\mathbf{K})$ **sejam todos negativos**.

A prova disto se encontra na bibliografia sobre controle e por ser um tanto longa não será apresentada neste capítulo. [8] O diagrama de blocos que representa a equação 3.14 se pode observar na figura 25

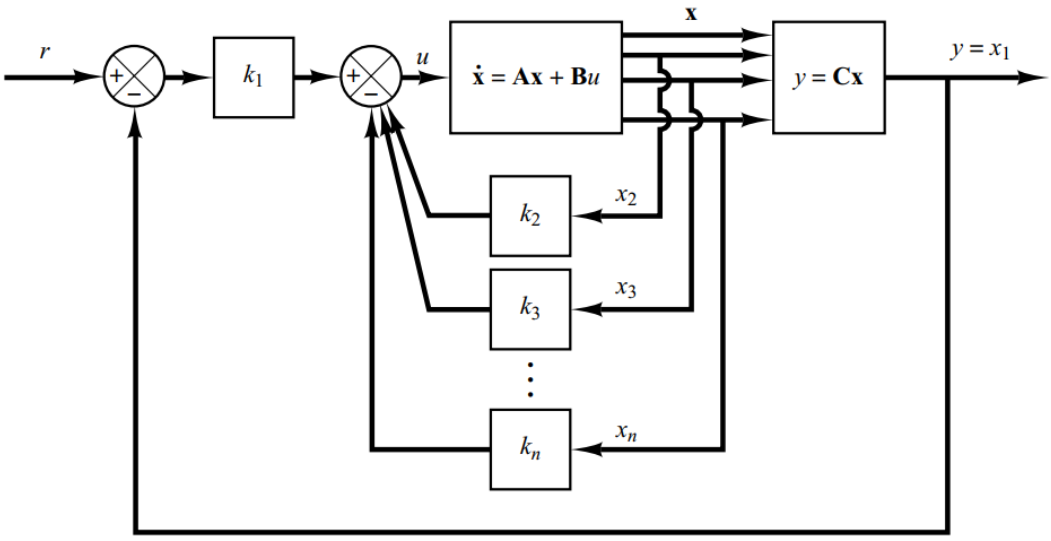


Figura 25 – Controle de sistemas SISO quando a planta é integradora [8]

Isto quer dizer que para sistemas integrativos, se pode usar apenas um ganho estático para garantir seguimento de degraus com erro zero em regime permanente. Se pode considerar esta razão, como análoga à explicação quando se tem uma planta (função de transferência) com polo na origem, uma realimentação com um ganho estático no erro garante o mesmo tipo de seguimento.

É possível aplicar os mesmos métodos, usados para estabilização de sistemas não integrativos, para calcular a matriz $\mathbf{K}$, como a alocação de polos, método LQR, etc.

3.4.2.2 Alocação de Polos

O primeiro método para calcular a matriz $\mathbf{K}$ apresentado é o método de alocação de polos. Ele é denominado assim pois usando os elementos desta matriz se pode alocar os autovalores (polos) da matriz $(\mathbf{A} - \mathbf{B}\,\mathbf{K})$, sempre e quando o sistema, representado pela equação, 3.12, seja **controlável**.

O sistema se diz controlável quando a matriz

$$\left[\mathbf{B} \quad | \quad \mathbf{AB} \quad | \quad \cdots \quad | \quad \mathbf{A^{n-1}B}\right], \tag{3.15}$$

tem posto completo, esta matriz é chamada de **matriz de controlabilidade**.

A forma mais comum de fazer alocar os polos é usando a formula de Ackermann. A formula se baseia em calcular os elementos da matriz de ganho tal que a seguinte equação seja verdade

$$|s\mathbf{I} - \mathbf{A} + \mathbf{BK}| = (s - \mu_1)(s - \mu_2)\cdots(s - \mu_n). \tag{3.16}$$

Onde $\mathbf{I}$ é uma matriz identidade de dimensão $n \times n$, e μ_i, para $i \in [1, ..., n]$, são os polos de malha fechada desejados. A algoritmo para calcular isto não sera discutido pois ele é longo, no entanto, muito conhecido na teoria de controle [8] [16]. Ainda, devido a dificuldade de usar o método teórico para um n grande, usualmente se usa algum programa de computador para calcular a matriz $\mathbf{K}$.

O problema do método de alocação de polos reside em que não se sabe se a dinâmica desejada se obtém com os polos escolhidos. A escolha dos polos geralmente se dá por intuição o que pode tornar o projeto deste tipo de controle trabalhoso, especialmente quando se tem varias malhas de controle.

3.4.2.3 Regulador Linear Quadrático

Este método surgiu como uma alternativa ao de alocação de polos. O Regulador Linear Quadrático, tem o mesmo esquema de controle ao usado na alocação de polos, mas sua diferencia reside em que ao invés de determinar a matriz para alocar os polos em lugares desejados, se liga seu calculo a um problema de minimização.

Dado um sistema do tipo

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \tag{3.17}$$

onde

$$\mathbf{u} = -\mathbf{K}\mathbf{x}. \tag{3.18}$$

para o qual se deseja minimizar o critério

$$J = \int_0^\infty (\mathbf{x}^*\mathbf{Q}\,\mathbf{x} + \mathbf{u}^*\mathbf{R}\,\mathbf{u})\,dt \tag{3.19}$$

onde $\mathbf{Q}$ e $\mathbf{R}$ são matrizes hermitianas definidas positivas (reais simétricas para $\mathbb{R}$).

As matrizes $\mathbf{Q}$ e $\mathbf{R}$ determinam a importância relativa da minimização dos estados e da magnitude do sinal de controle, respectivamente. Colocando de forma mais simples, $\mathbf{Q}$ e $\mathbf{R}$ ponderam o que é mais importante, a velocidade com a qual a dinâmica do vetor de estados vai para zero, ou, minimizar a energia no sinal de controle para garantir a estabilização do sistema.

Assim como no caso do método de alocação de polos, o algoritmo usado para calcular a matriz de ganho ótima, não será apresentada, pois a explicação da sua logica é longa. Também este método é facilmente implementado em um programa. A maioria dos livros de controle moderno possui a logica aplicada para o calculo da matriz de ganhos [8] [16].

O problema do LQR esta na forma de ponderar corretamente os valores das matrizes $\mathbf{Q}$ e $\mathbf{R}$, também depende da intuição do projetista, mas, ao contrario da alocação de polos, se pode tomar em conta não só a dinâmica da estabilização, mas também limitar a energia do controle.

3.5 Resumo do Capítulo

Neste capítulo se fez uma definição mais completa do problema que o sistema de controle deve resolver. Este se trata de um problema de seguimento de trajetória por um robô, onde a ela é definida pelo Algoritmo de Movimento. A plataforma deve funcionar **idealmente** na faixa de 0 a 10 Hz, mas no **mínimo** 1 Hz. A trajetória deve ser seguida em no **máximo** 2 s, **idealmente** 1 s, sem sobre sinais e respeitando as limitações físicas dos atuadores.

Se fez uma descrição cinemática da plataforma, com foco na **cinemática inversa**, que será necessária para implementar a solução. Isto se deve à **estrutura de controle no espaço das juntas**, que se demostro ser a mais simples de ser implementada.

A estrutura não necessariamente determina o tipo de controlador a ser usado e portanto se apresentou quatro métodos para projetar os controladores, sintonização PID, lugar das raízes, alocação de polos e LQR. No caso dos dois ultimo se ressaltou o seu no caso especial no qual existe um integrador no sistema a ser controlado, que se aplica neste projeto.

4 Métodos Adicionais Usados

Em este capítulo serão apresentados outros fundamentos teóricos que também foram necessários para implementar os diferentes controladores. Estes embasamentos teóricos foram indispensáveis devido à problemas observados nos testes iniciais para implementação dos controladores. Os problemas encontrados não serão discutidos detalhadamente neste capítulo mas se dará uma explicação superficial da razão da necessidade destes métodos.

O primeiro impasse para realizar os testes é a má qualidade do sinal que se tinha dos transdutores das juntas prismáticas (potenciômetros), em virtude do ruído presente nele. Isto pode ser resolvido usando a teoria de filtragem de sinais e projetando um **Filtro de Medida**.

Outra nuança encontrada era a falta de um modelo matemático simples para projetar os controladores, uma vez que, todos os controladores considerados precisam deste modelo para serem projetados. A solução considerada foi realizar uma identificação da resposta de cada perna a uma certa entrada, assim foi necessária a teoria de **Identificação de Sistemas**.

Uma pequena ressalva existe ao usar controladores com a realimentação de estados, é necessário um modelo do sistema em espaço de estados para poder projetar eles. Assim será explicado uma metodologia usada para realizar a **Transformação para o Espaço de Estados**.

Por último, para usar especificamente as metodologias de controle por realimentação de estados (alocação de polos e LQR), é necessário medir os estados do sistema, como foi explicado no capítulo anterior. Como esta medida não existe na prática é preciso estimar os estados através de um **Observador de Estados**

4.1 Filtro de Medida

Usualmente quando se trabalha com sistemas reais existe o problema de aparecer interferência e ruído nos sinais internos usados. Geralmente a causa disto é devido aos vários campos eletromagnéticos que se tem ao redor dos meios pelos quais os sinais são transmitidos. Por exemplo, correntes que passam por um fio elétrico, geram campos magnéticos que interatuam com outros fios.

As causas podem ser diversas e muitas vezes difíceis de especificar, o que torna pouco prático tentar evitar qualquer tipo de interferência na maioria dos casos. Uma solução prática é o uso de filtragem de sinais.

É denominado deste modo pois se explota a diferença de frequências entre o sinal original e outros sinais que o corrompem, para "filtrar" o sinal das suas impurezas, análogo a como se filtra a água.

Geralmente quando se projeta algum sistema se tem uma ideia das qualidades dos sinais que serão usado, uma delas é a sua banda, que se refere ao intervalo de frequências no qual se espera que o sinal funcione.

Assim filtros ideais seriam funções da frequência do sinal ($H(\omega)$), tais que, para as frequências desejadas teriam um ganho unitário, com intuito de não mudar o sinal. Já, para sinais de frequências não desejadas, se usaria ganho zero para filtrar eles de forma absoluta.

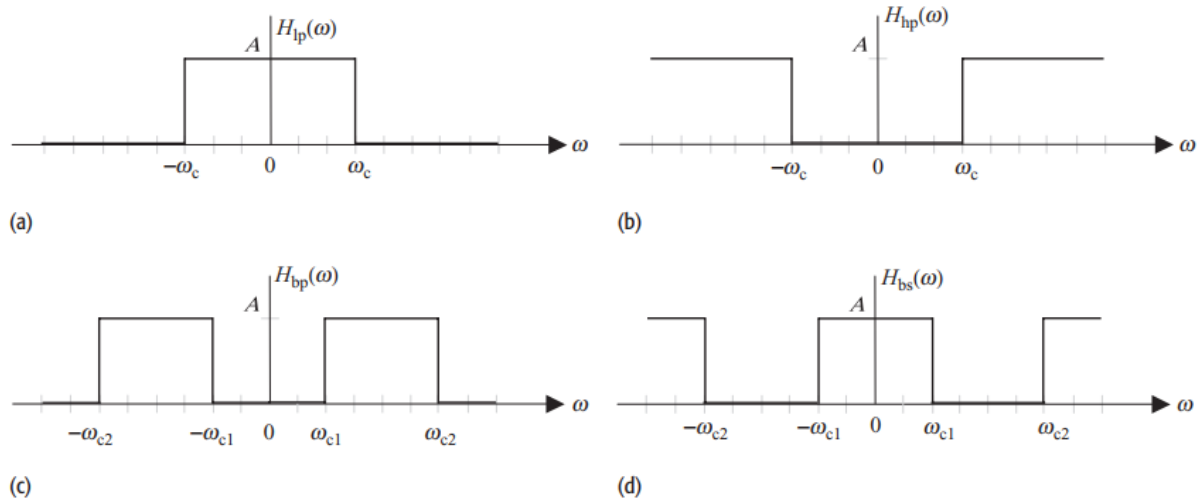


Figura 26 – Resposta na frequência de quatro tipos de filtros ideais [9]

O conjunto das frequências que se desejam rejeitar se chama de **banda rejeitada** (**stop band**) e o conjunto de frequências que se desejam manter se denomina banda passante (**pass band**). Os tipos de filtros se podem observar na figura 26, eles são classificados da seguinte forma:

(a) Filtro passa-baixa: Onde a banda passante se encontra nas frequências baixas, ou seja, a partir do frequência 0 até uma frequência determinada;

(b) Filtro passa-alta: Onde a banda passante se considera de alta frequência, ou seja, a partir de uma determinada frequência até a máxima frequência (frequência infinita);

(c) Filtro passa-faixa: Onde banda passante se encontra em uma certa faixa de frequências e o resto das frequências é filtrada.

(d) Filtro rejeita-faixa: Ao revés do passa faixa, este filtro rejeita uma determinada faixa de frequências enquanto deixa passar o resto.

Contudo, implementar este tipo de filtros em tempo real, seria impraticável [9], pois não é possível ter ganhos constantes para um certa banda além de ocorrer uma transição abrupta entre as bandas.

Assim é inevitável relaxar as características que se querem no filtro, como no caso de, ao invés de, ter um ganho constante na banda passante, ter uma ganho que varia numa certa faixa ($\pm\delta_p$). Da mesma forma, para a banda rejeitada se considera que o ganho pode variar dentro de uma faixa (δ_s), como se descreve na equação a seguir

$$\begin{aligned} 1-\delta_p \leq |H(\omega))| \leq 1+\delta_p \quad &\text{,para } \omega \text{ na banda passante,} \\ 0 \leq |H(\omega))| \leq 1+\delta_s \quad &\text{,para } \omega \text{ na banda rejeitada.} \end{aligned} \tag{4.1}$$

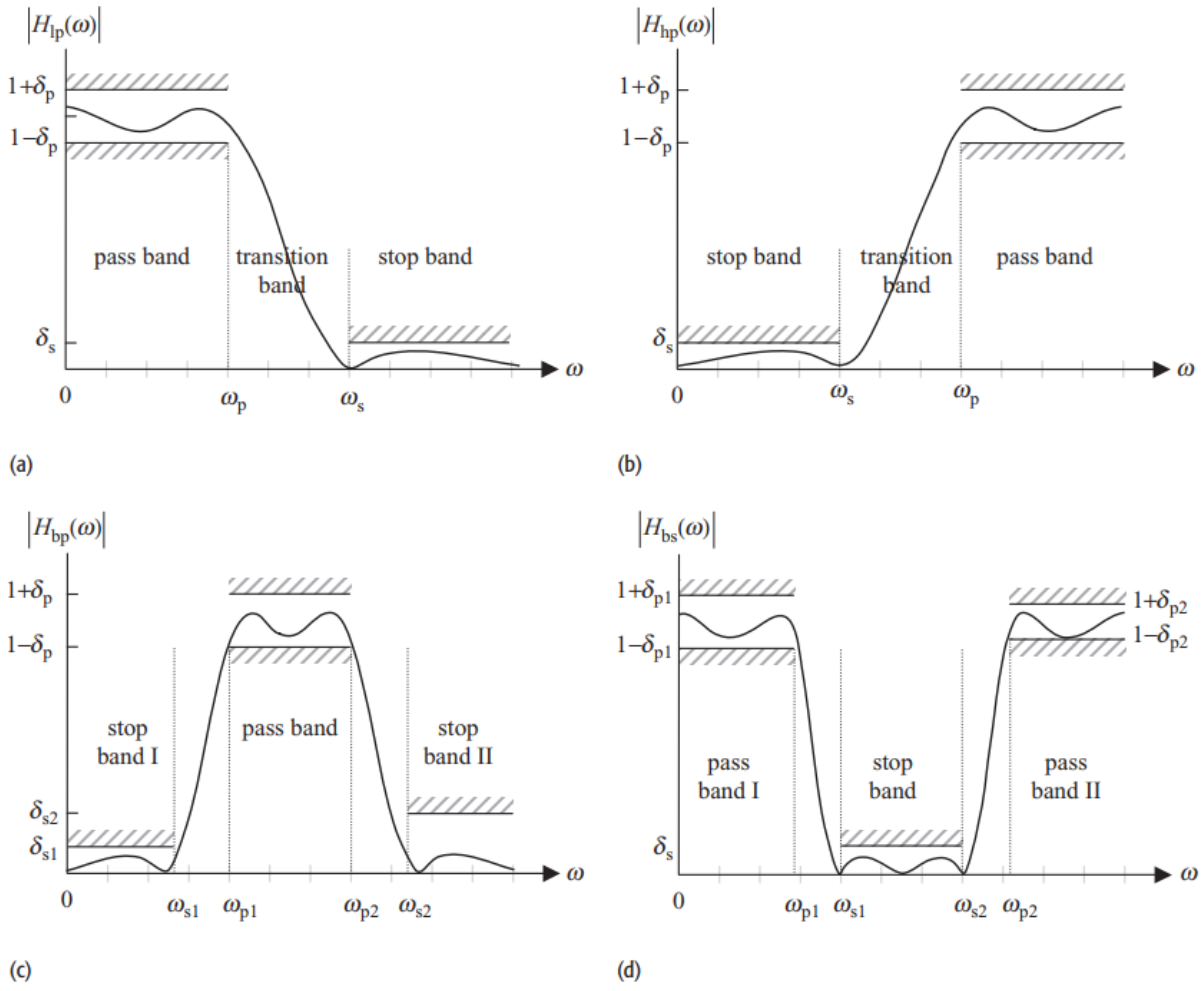


Figura 27 – Resposta na frequência de quatro tipos de filtros reais [9]

A estas "folgas" se denominam ***ripples***. Outra característica que se deve relaxar é a mudança abrupta entre as duas bandas, agregando uma banda intermediaria, denominada **banda de transição** (***transitional band***). Esta é delimitada pelas **frequências de canto** (***corner frequency***), na banda passante (ω_p) e na banda rejeitada (ω_s). Com estas características os filtros ideais da figura 26 passam a ser os que se vem na figura 27.

Devido à adição da banda de transição, se usa uma frequência denominada de **frequência de corte** (***cut-off***), onde se tem um ganho de -3 dB, aproximadamente 0.707.

Em essência funções de transferência são filtros, pois cada uma delas tem uma resposta particular na frequência, que depende dos seus polos, zeros e do ganho. Idealmente se quer, para funções de filtros, que o ganho estático seja unitário, de modo a garantir que em regime permanente o sinal não mude de escala.

Neste projeto eram necessários filtros passa baixa e portanto as teoria apresentadas a seguir serão apenas para este caso. O filtro passa baixa mais simples de ser implementado é o seguinte

$$H(\omega) = \frac{1}{\dfrac{\omega}{\omega_c} + 1}, \tag{4.2}$$

onde ω_c é a frequência de corte, em rad/s, que esta função de transferência terá.

Se pode ainda classificar os filtros com base nas características da função de transferência e sua relação com sua resposta na frequência.

4.1.1 Tipos de Filtro Passa Baixa

Os filtros apresentados a seguir são os mais conhecidos e usados. A explicação da lógica usada para obter as características desejadas é muito longa e portanto apenas se apresentará a sua relação com a magnitude da resposta na frequência. As informações completas se encontram na bibliografia [9].

(a) **Filtro Butterworth**: É o tipo mais simples, onde com a informação da frequência de corte desejada e da ordem (N) da função de transferência, se projeta o filtro onde se tenha a seguinte resposta na magnitude

$$|H(\omega)| = \frac{1}{\sqrt{1 + \left(\frac{\omega}{\omega_c}\right)^{2N}}}. \tag{4.3}$$

Este tipo não tem um *ripple* considerável na banda passante e na rejeitada, mas tem a desvantagem de possuir uma banda de transição relativamente grande, em comparação ao outros. A função de transferência deste filtro é dada por

$$H(s) = \frac{1}{\prod_{n=1}^{N} (s - p_n)}. \tag{4.4}$$

(b) **Filtro Chebyshev Tipo I**: Este filtro possui uma banda de transição menor que a do Butterworth, mas possui ripple na banda passante. A sua resposta na frequência para a magnitude é dada por

$$|H(\omega)| = \frac{1}{\sqrt{1 + \varepsilon^2 {T_N}^2 (\omega/\omega_p)}}. \tag{4.5}$$

onde T_N é o polinômio de Chebyshev de grau N, o qual é

$$T_N(\omega) = \begin{cases} \cos(N \cos^{-1}(\omega)) & |\omega| \leq 1 \\ \cosh(N \cosh^{-1}(\omega)) & |\omega| > 1 \end{cases}. \tag{4.6}$$

O projeto analítico deste filtro é difícil, então se usa algum programa que calcule a função de transferência para uma dada ordem, frequência de corte e ripple máximo, que se permite na banda passante.

(c) **Chebyshev Tipo II**: Também chamado de filtro Chebyshev inverso, pois possui a mesma característica na banda de transição, no entanto o ripple aparece na banda passante. Sua resposta na frequência para a magnitude é

$$|H(\omega)| = \frac{1}{\sqrt{1 + \left[\varepsilon^2 {T_N}^2 (\omega/\omega_p)\right]^{-1}}} = \sqrt{\frac{\varepsilon^2 {T_N}^2 (\omega/\omega_p)}{1 + \varepsilon^2 {T_N}^2 (\omega/\omega_p)}}. \tag{4.7}$$

Igualmente se usa o polinômio de Chebyshev, e por consequência é mais pratico fazer o projeto deste filtro usando um *software*.

(d) **Filtro Elíptico**: Este filtro possui a menor faixa de transição, entre os vistos, mas, por conta disto, existe ripple dentro da banda passante e rejeitada. Ele é dado por

$$|H(\omega)| = \frac{1}{\sqrt{1 + \varepsilon^2 {U_N}^2(\omega)}}. \tag{4.8}$$

onde $U_N(\omega)$ é a função elíptica Jacobiana de grau N (mais informações em [9]). Usualmente também se evita projetar ele analiticamente.

A figura 28, evidencia as diferenças existentes entre os tipos de filtros passa baixa descritos. Também se pode observar as características descritas neste capítulo, como ω_p e ω_s, frequência de canto da banda passante e da banda rejeitada, respectivamente; e δ_p e δ_s, *ripple* da banda passante e da banda rejeitada, respectivamente. Estas características serão necessárias para realizar o projeto dos filtro posteriormente.

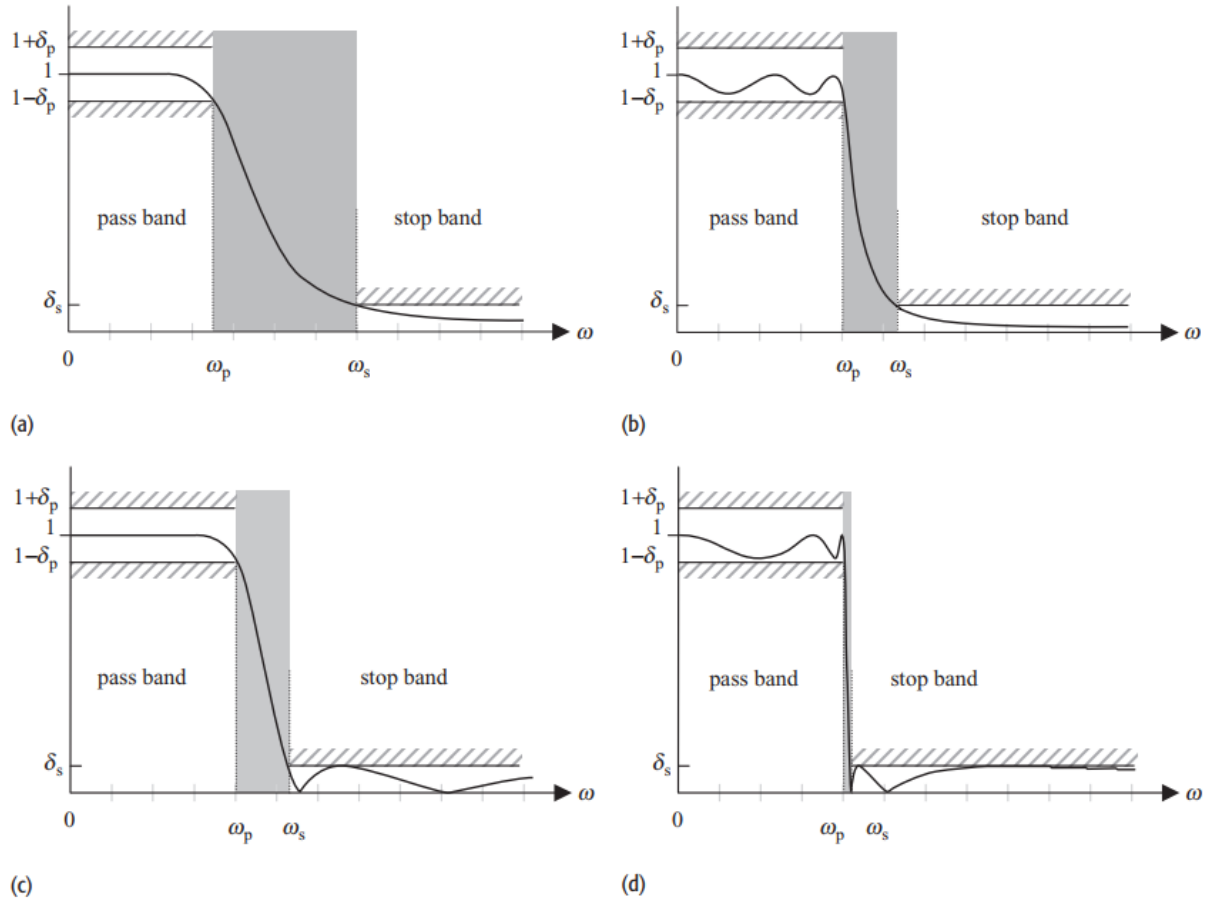


Figura 28 – Comparação da resposta na frequência entre os tipos de filtro passa baixa [9]

4.2 Identificação de Sistemas

Para poder projetar a maioria dos controladores é preciso ter um modelo da planta com o qual se baseia o cálculo dos parâmetros dele. Existem metodologias de projeto de controlador, onde não se requere o modelo, como alguns métodos de sintonização PID, onde apenas se observa as características da saída para determinar os ganhos. Contudo se deseja projetar outros tipos de controladores, e portanto é necessário fazer uma modelagem da planta.

Primeiramente se ponderou usar uma linearização do modelo dinâmico da plataforma, no entanto, devido à dificuldade em mensurar certas características deste modelo, se optou por fazer uma identificação, observando o comportamento da saída de cada junta.

Usualmente se faz identificação de sistemas usando a reposta ao degrau, mas devido à peculiaridade do mecanismo de movimento, já que se trata de um sistema integrativo, não é possível usar este método. Isto decorre da característica de sistemas integrativos, que faz com que a sua resposta ao degrau seja uma rampa.

Poderia se calcular a derivada deste sinal, mas devido à má qualidade do sinal, em razão da presença do ruído de medida, isto não era favorável. Lembrando que quando

se deriva um sinal o ruído também é derivado, piorando ainda mais o ruído. Se poderia contornar o problema usando uma entrada impulso, ao invés de degrau de forma a obter uma resposta degrau na saída, mas implementar na prática um impulso é impossível e mesmo assim existiria o problema da má qualidade do sinal medido.

Nos testes iniciais se observou que a qualidade da medida melhorava quando existia movimento na plataforma, como no caso da entrada degrau. Se verificou experimentalmente que o ruído dos transdutores prevalecia mais quando a medida ficava "estável", ao contrário quando se tinha um movimento de velocidade constante. Logo se teorizo um método para poder identificar sistemas através da resposta rampa.

4.2.1 Identificação com Resposta Rampa

A identificação de sistemas quando se tem uma rampa como entrada não é muito habitual, e devido a isso não se achou uma bibliografia que ajudasse neste aspecto. No entanto o aluno, com apoio do orientador na UNESP, criou um método de identificação de sistemas quando se tem dados da resposta a uma entrada rampa. O método se baseia na analise matemática da resposta no tempo do sistema quando se tende ao infinito.

Dada uma entrada rampa

$$U(s) = \frac{k_e}{s^2}, \tag{4.9}$$

com ganho k_e, aplicada ao seguinte sistema

$$G(s) = \frac{Y(s)}{U(s)} = \frac{k_g}{(\tau_1 s + 1)(\tau_2 s + 1)}. \tag{4.10}$$

Onde k_g é o ganho estático do em regime permanente e $[\tau_1, \tau_2]$ são as constantes de tempo dos polos, da função de transferência.

Dado isto pode se calcular a saída do sistema a uma dada entrada, da seguinte forma

$$Y(s) = G(s) \cdot U(s). \tag{4.11}$$

Usando a equação 4.11 e substituindo 4.9, se pode achar esta a saída no tempo usando a transformada inversa de Laplace.

$$y(t) = \mathcal{L}^{-1}\{Y(s)\} = \mathcal{L}^{-1}\left\{\frac{k_g}{(\tau_1 s + 1)(\tau_2 s + 1)} \cdot \frac{k_e}{s^2}\right\}, \tag{4.12}$$

$$y(t) = k_g k_e \left(t - (\tau_1 + \tau_2) + \frac{{\tau_1}^2}{\tau_1 - \tau_2} e^{-t/\tau_1} + \frac{{\tau_2}^2}{\tau_2 - \tau_1} e^{-t/\tau_2}\right). \tag{4.13}$$

Usando a equação 4.13, se pode analisar o comportamento da saída quando o tempo tende a zero.

$$\lim_{t\to\infty} y(t) = k_g k_e \left(\lim_{t\to\infty} t - (\tau_1 + \tau_2) + \lim_{t\to+\infty} \left(\frac{{\tau_1}^2}{\tau_1 - \tau_2} e^{-t/\tau_1} + \frac{{\tau_2}^2}{\tau_2 - \tau_1} e^{-t/\tau_2} \right) \right), \tag{4.14}$$

$$\lim_{t\to\infty} y(t) = \lim_{t\to\infty} \left(k_g k_e \, t - k_g k_e (\tau_1 + \tau_2) \right). \tag{4.15}$$

A partir da equação 4.15 se pode concluir que, a medida que o tempo avança, o comportamento da saída se assemelha à uma função de 1º grau, do tipo

$$y(t) = a\, t + c, \tag{4.16}$$

onde a é o coeficiente angular da reta e c é o coeficiente linear da reta. Isto é verdade para qualquer função de transferência $G(s)$ de 2ª Ordem com polos reais, pois a medida que passa o tempo a influencia, na saída $Y(s)$, das exponenciais da equação 4.13 vá diminuindo.

Pode considerar-se que o regime "permanente" do sistema acontece quando o comportamento da saída é aproximadamente linear, pois ele se aproxima assintoticamente à reta

$$f(t) = k_g k_e \, t - k_g k_e (\tau_1 + \tau_2). \tag{4.17}$$

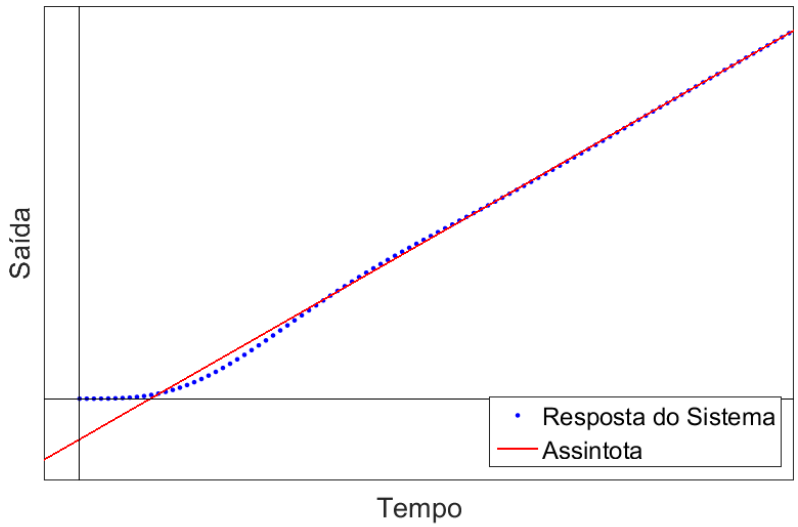


Figura 29 – Resposta de um sistema genérico à uma entrada rampa

Caso se tenha disponíveis dados suficientes da resposta à rampa do sistema, a reta que representa a assíntota mostrada na figura 29 pode ser estimada usando um método matemático ou visual.

Sendo a reta $\bar{A}$, a que melhor define a assíntota, e assumindo que se conhecem os valores do seu coeficiente angular (α) e coeficiente linear (t_0), definida pela equação

$$\bar{A} = \alpha\, t + t_0, \tag{4.18}$$

Igualando as equações 4.18 e 4.17, se pode calcular k_g facilmente, sempre e quando se conheça k_e, pois

$$k_g = \frac{k_e}{\alpha}. \tag{4.19}$$

Para calcular as constantes de tempo falta informações pois se tem apenas uma equação e duas incógnitas,

$$\tau_1 + \tau_2 = \frac{t_0}{\alpha}. \tag{4.20}$$

Uma forma de resolver é fazendo um palpite educado de uma das constantes e calcular a outra, e, com o uso de simulações, recursivamente melhorar este palpite. Outra forma seria assumir que ambas constantes de tempo são iguais, ou seja, que o sistema é criticamente amortecido, desta forma a equação 4.20 pode ser simplificada e se pode obter o seguinte

$$\tau = \frac{t_0}{2\alpha}. \tag{4.21}$$

Com isto o sistema $\hat{G}(s)$ identificado tem a seguinte função de transferência

$$\hat{G}(s) = \frac{k_g}{(\tau\, s + 1)^2} \approx G(s). \tag{4.22}$$

Este método usado poderia ser generalizado para qualquer tipo de sistema linear, porém com apenas a teoria demostrada já é suficiente para implementar a solução descrita nos capítulos posteriores.

4.2.1.1 Aplicação para Sistemas Integrativos

A teoria discutida se aplica para entradas rampa, mas se pode aplicar a mesma logica para entrada degrau em sistemas integrativos como no caso da plataforma.

Assumindo que a plataforma trabalha numa região onde o comportamento dela é relativamente linear, se pode dividir o modelo dela em duas partes, uma parte seria

aproximada por uma função de transferência $G(s)$ de segunda ordem, a qual se quer identificar, a outra é a ação integrativa presente na planta, que se modela como se fosse o comportamento do potenciômetro, como se pode ver no diagrama de blocos da figura 30.

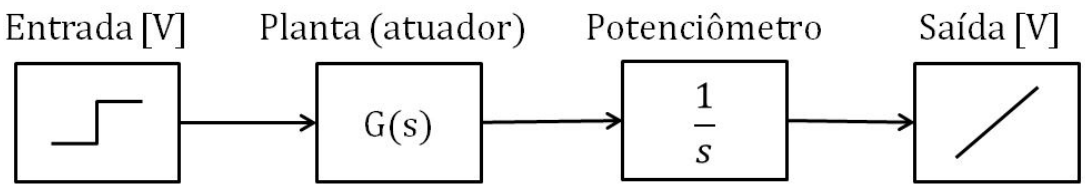


Figura 30 – Modelo proposto para aproximar a dinâmica de uma junta ativa

Como se assume que o sistema é linear, colocar a ação integrativa antes de $G(s)$ não deveria afetar o sistema total, com isso pode-se agrupar a entrada degrau junto com a ação integrativa, para formar uma rampa. Isto é exemplificado na figura 31. Assim se pode usar a técnica de identificação com entrada rampa discutida anteriormente.

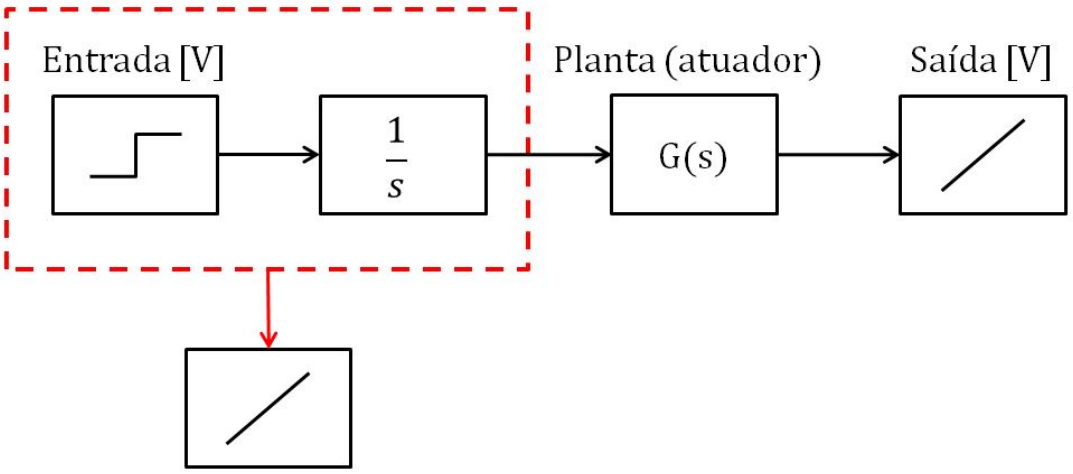


Figura 31 – Lógica proposta de agrupamento de entrada e ação integrativa

4.3 Transformação para o Espaço de Estados

Como foi discutido na seção anterior, o melhor forma de modelar o dinâmica das juntas da plataforma seria usando uma identificação por funções de transferência da medida do transdutor para uma dada entrada no motor.

No entanto na seção 3.4.2, se usa técnicas que necessitam de um modelo no espaço de estados. Então é preciso realizar uma transformação dos modelos no domínio da frequência para o espaço de estados. Além disto o modelo usado deve atender os requisitos determinados na seção 3.4.2.1. É imprescindível que o modelo tenha como saída o primeiro estado para poder aplicar estas técnicas, ou seja, $y(t) = \mathbf{C}\,\mathbf{x}(t) = x_1(t)$.

A metodologia para realizar esta transformação e atender este requisito será apresentada a seguir. Dada uma função de transferência genérica, que determina o comportamento para um sistema com entrada $u(t)$ e saída $y(t)$,

$$\frac{Y(s)}{U(s)} = \frac{b}{s^n + a_1 s^{n-1} + \cdots + a_{n-1}s + a_n}. \tag{4.23}$$

O numero de estados deste sistema é igual ao maior expoente que a variável s possui, neste caso n. Para realizar a transformação é preciso determinar a dinâmica dos estados, através um sistema de equações diferencias, tal que

$$\begin{aligned} \dot{x}_1 &= \alpha_{11}x_1 + \cdots + \alpha_{1n}x_n + \beta_1 u, \\ &\vdots \\ \dot{x}_n &= \alpha_{n1}x_1 + \cdots + \alpha_{nn}x_n + \beta_n u. \end{aligned} \tag{4.24}$$

Transformando a equação 4.23 e transformado para o domínio do tempo se obtém

$$\overset{(n)}{y}(t) + a_1 \overset{(n-1)}{y}(t) + \cdots + a_{n-1}\dot{y}(t) + a_n y(t) = b\,u(t). \tag{4.25}$$

Determina-se que a derivada dos primeiros $n-1$ estados é o estado seguinte, e sabendo que $y(t) = x_1(t)$, pode-se concluir que

$$\begin{aligned} \dot{y}(t) &= \dot{x}_1(t) = x_2(t) \\ \ddot{y}(t) &= \dot{x}_2(t) = x_3(t) \\ &\vdots \\ \overset{(n-1)}{y}(t) &= \dot{x}_{n-1}(t) = x_n(t) \end{aligned} \tag{4.26}$$

Entretanto ainda é necessário determinar a dinâmica do ultimo estado, isto se pode realizar usando as equação 4.25 e substituindo as variáveis com o sistema de equações 4.26 e isolando $\dot{x}_n(t)$. Assim se obtém

$$\overset{(n)}{y} = \dot{x}_n = b\,u - a_n x_1 - a_{n-1}x_2 - \cdots - a_2 x_{n-1} - a_1 x_n \tag{4.27}$$

Finalmente se transforma notação usada para a forma matricial e se pode concluir

que a dinâmica dos estados é regida pela seguinte equação

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \vdots \\ \dot{x}_{n-1} \\ \dot{x}_n \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & a_1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ b \end{bmatrix} u. \tag{4.28}$$

Uma vez que $y(t) = \mathbf{C}\,\mathbf{x}(t) = x_1(t)$, se deduz que $\mathbf{C} = \begin{bmatrix} 1 & \cdots & 0 \end{bmatrix}$, assim o sistema se pode escrever de forma reduzida

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \mathbf{A}\,\mathbf{x}(t) + \mathbf{B}u(t) \\ y(t) &= \mathbf{C}\,\mathbf{x}(t). \end{aligned} \tag{4.29}$$

Com isto se garante o requisito e se conseguiu transformar ao espaço de estados a função de transferência inicial. Esta transformação é similar forma canônica de controlabilidade que se conhece na literatura [8], exceto que as matrizes $\mathbf{B}$ e $\mathbf{C}$ são diferentes. Este método também só possível de ser usado para modelos onde os estados dependem apenas da controle atual, isto é, a função de transferência não possui zeros.

4.4 Observador de Estados

As técnicas de controle moderno mais comuns são realizadas com a realimentação dos estados do sistema, no entanto, não é usual que todos os estados do sistema sejam medidos. Usualmente se mede apenas um dos estados, ou uma combinação linear deles, o qual se deseja controlar ($y(t) = \mathbf{C}\mathbf{x}(t)$). As razões disto são várias, mas geralmente se torna pouco prático realizar medidas de todos os estados. Para contornar este problema se usam elementos de controle denominados de **Observadores de Estado** ou simplesmente **Observadores**. Eles se encarregam em estimar os estados do sistema, através de um calculo matemático, com o uso da variável medida do sistema $\mathbf{y}(t)$.

Dado um sistema definido pela equação 4.29. Poderia se estimar os estados apenas invertendo o sistema, em função das variáveis de saída e de entrada, para calcular os estados, desde que as matrizes $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{C}$ sejam inversíveis, o que normalmente é falso para sistemas MIMO e sempre falso para SISO ($\mathbf{C}$ é uma matriz linha).

Contudo, ainda que se consiga realizar este calculo, é provável que as matrizes que determinam a dinâmica do sistemas sejam apenas uma aproximação matemática, isto é, se admite que elas podem divergir do sistema real, o que acarretaria um erro nos estados estimados.

Assim, para realizar a estimação dos estados $\tilde{\mathbf{x}}(t)$, se usa um erro de observação para resolver este problema. Este erro seria calculado pela diferença entre a saída medida e a saída estimada $\tilde{y}(t) = \mathbf{C}\tilde{\mathbf{x}}(t)$, e multiplicado por uma matriz constante $\mathbf{K}_e$ que se tem liberdade de determinar. Determina-se assim a a dinâmica dos estados estimados como sendo

$$\begin{aligned}\dot{\tilde{\mathbf{x}}}(t) &= \mathbf{A}\tilde{\mathbf{x}}(t) + \mathbf{B}u(t) + \mathbf{K}_e(y(t) - \mathbf{C}\tilde{\mathbf{x}}(t)) \\ &= (\mathbf{A} - \mathbf{K}_e\mathbf{C})\tilde{\mathbf{x}}(t) + \mathbf{B}u(t) + \mathbf{K}_e y(t).\end{aligned} \tag{4.30}$$

Neste caso se estimam todos os estados do sistema, sendo assim o observador é classificado como um observador de estados de ordem completa. O diagrama de blocos deste sistema é observado na figura 32.

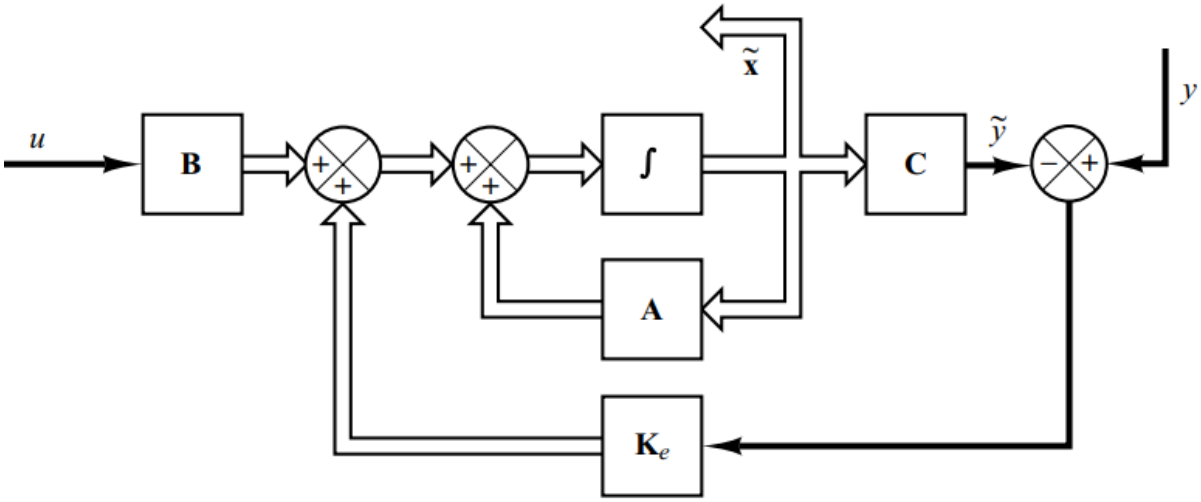


Figura 32 – Diagrama de blocos de um observador de estados de ordem completa [8]

Pode-se provar que existe uma condição na qual o erro entre os estados reais e os estimados converge assintoticamente para zero neste sistema. Para isto se faz a diferença entre as equações 3.12 e 4.30, se obtém assim

$$\dot{\mathbf{x}} - \dot{\tilde{\mathbf{x}}} = (\mathbf{A} - \mathbf{K}_e\mathbf{C})(\mathbf{x} - \tilde{\mathbf{x}}). \tag{4.31}$$

Definindo que $\mathbf{e} = \mathbf{x} - \tilde{\mathbf{x}}$, como um vetor de erro, se obtém

$$\dot{\mathbf{e}} = (\mathbf{A} - \mathbf{K}_e\mathbf{C})\mathbf{e}, \tag{4.32}$$

se pode concluir que se a matriz $\mathbf{A} - \mathbf{K}_e\mathbf{C}$ é estável, então os estados estimados convergem assintoticamente para o valor dos estados reais, e que a velocidade de convergência depende dos autovalores desta mesma matriz.

Como se tem a liberdade de escolher a matriz $\mathbf{K}_e$, denominada de matriz de ganho do observador, se pode provar que é possível alocar os polos deste sistema usando a mesma técnica descrita na seção 3.4.2.2, para o sistema dual $\mathbf{A}^T - \mathbf{C}^T\mathbf{K}_e$. Igualmente, isto é possível sempre e quando o sistema dual seja controlável, ou em termos dos sistema original, que este seja **observável**.

A observabilidade de um sistema dado pela 3.12 é determinada, pelo posto da **matriz de observabilidade**, a qual é definida como

$$\begin{bmatrix} \mathbf{C} & | & \mathbf{CA} & | & \cdots & | & \mathbf{CA^{n-1}} \end{bmatrix}^T. \tag{4.33}$$

Similar à propriedade de controlabilidade descrita anteriormente, um sistema é observável se e somente se a matriz de observabilidade tenha posto completo, ou seja, posto igual ao número de estados do sistema.

Na literatura, se quer que o observador de estados, projetado para um sistema de controle, consiga estimar os estados com uma velocidade 2 a 5 vezes maior que a velocidade de resposta do sistema de controle, para evitar que a dinâmica do observador afete o funcionamento planejado do controlador [8]. Considerando que se tenha apenas aproximações das matrizes $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{C}$ do sistema real, se deve tomar cuidado na hora de projetar o matriz de ganho do observador, pois os erros da aproximação podem causar que o observador tenha um erro constante em regime permanente ou, no pior dos casos, não seja estável.

4.5 Resumo do Capítulo

Neste capítulo se discutiu alguns métodos adicionais que foram necessários para a implementação das soluções do problema.

A necessidade de um filtro de medida foi devido à natureza ruidosa dos sinais dos transdutores nas juntas, os quais eram a única forma de realizar alguma estimativa da posição e orientação da plataforma, como foi mencionado na seção 2.3 e na seção 2.4. Assim se apresentaram 4 tipos de filtro passa-baixa mais usados na seção 4.1.

Para projetar alguns dos controladores usando as metodologias vistas na seção 3.4, é primordial ter um modelo da planta. Como demostrado na seção 4.2, devido à característica ruidosa do sinal e à particularidade dos sistemas integrativos, não é viável usar o método usual de identificação à resposta degrau.

O aluno, com a orientação do supervisor, determinou uma lógica para fazer uma identificação de sistemas, usando a resposta rampa, para determinar uma função de transferência de 2ª ordem. Este método também era possível de ser aplicado a sistemas integrativos, como no caso do sistema de movimento.

Como foi visto na seção 3.14, para realizar o projeto dos controladores é necessário um modelo no espaço de estados que garanta que $y(t) = x_1(t)$, portanto é necessário usar uma transformação que garanta isto. A metodologia usada para isto é explicada na seção 4.3, ela é similar à transformação na forma canônica de controlabilidade, com algumas diferenças na matrizes $\mathbf{B}$ e $\mathbf{C}$.

Como foi visto na seção 3.4.2, o uso de realimentação de estados é vital ter medidas de todos os estados do sistema, que usualmente é falso, como no caso deste sistema de controle. Assim na seção 4.4 se apresenta o conceito de observador de estados que se encarrega de estimar os estados do sistema usando a saída medida e o sinal de controle que esta sendo usado. Usualmente para uso em malhas de controle fechado se quer que o observador seja 2 a 5 vezes mais rápido.

5 Projeto do Sistema de Controle

No capítulo a seguir será discutido o projeto dos controladores para resolver o problema, descrito na seção 3.1, de seguimento de trajetória da plataforma de Stewart, tomando em conta os conceitos teóricos expostos nos capítulos 3 e 4.

Antes de poder projetar o controlador que irá solucionar o problema em questão é fundamental definir duas características, que limitam o projeto do controlador, como foi visto nas seções 3.3 e 4.2.

A primeira é a **estrutura de controle**, pois a forma como será estruturado o sistema de controle, limita os tipos de controladores a serem usados. A segunda é o **modelo do planta** usado para fazer o projeto dos controladores. Assim mesmo, este restringe o método usado para o projeto do controlador. Sendo assim, estes dois serão debatidos no inicialmente neste capítulo nas seções 5.1 e 5.2.

Antes de poder realizar o projeto é necessário sanar o problema do ruído encontrado nos transdutores que medem o deslocamento das juntas prismáticas do robô. Isto é vital, pois as estrategias de controle definidas precisarão de um sinal de boa qualidade destes sensores. Como foi visto no capítulo 4, sabendo-se a natureza do ruído, pode-se projetar um filtro que consiga melhorar a qualidade dos sinais realimentados.

Com todas estas características definidas, finalmente se pode realizar o projeto dos controladores, contudo alguns destes métodos necessitam de outros elementos que serão descritos também.

5.1 Estrutura de Controle Usada

Entre as estruturas de controle discutidas na seção 3.3, considerou-se o **Controle no Espaço das Juntas**, o mais adequado para solucionar o problema, pois, comparado com o controle baseado na dinâmica inversa, é mais simples de ser implementado, já que não é necessário o uso de controladores não lineares, além de não ser necessário a realimentação da derivada dos sinais dos transdutores das juntas ativas.

O controle no espaço cartesiano não é possível de ser implementado, uma vez que apenas se pode medir as coordenadas cartesianas do robô com a IMU, e enviar elas por comunicação serial. Isto impediria o uso concorrente do Sistema de Movimento com o Modelo Dinâmico da Aeronave, como foi exposto no capítulo 3. Se poderia contornar este problema usando as medidas no espaço das juntas e transformá-las ao espaço cartesiano, como se pode observar na figura 33. Onde o *Filtro Washout* reapresenta o algoritmo de controle.

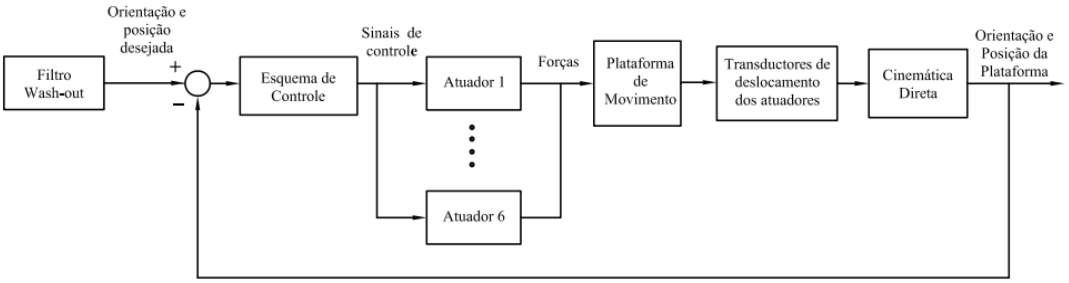


Figura 33 – Estrutura do controle no espaço cartesiano proposta [3]

Todavia, como foi mencionado na seção 3.2, a resolução da cinemática inversa para a plataforma de Stewart, não é trivial e difícil de se implementar em tempo real, o que descarta esta solução. Restando, entre as estruturas estudadas, apenas o uso da estrutura de controle no espaço das juntas.

5.1.1 Estrutura de Controle Adotada

Como foi vista na seção 3.3, o controle no espaço das juntas faz uso da cinemática inversa para converter a referência de posição e orientação (espaço cartesiano), que o robô deve seguir, para uma referência de comprimento dos atuadores (espaço das juntas), equivalente a ela.

Desta maneira se pode usar a realimentação dos comprimentos dos atuadores para calcular o erro do seguimento e usar este no cálculo da ação de controle para corrigir o movimento da plataforma. A estrutura usada pode ser observada na figura 34.

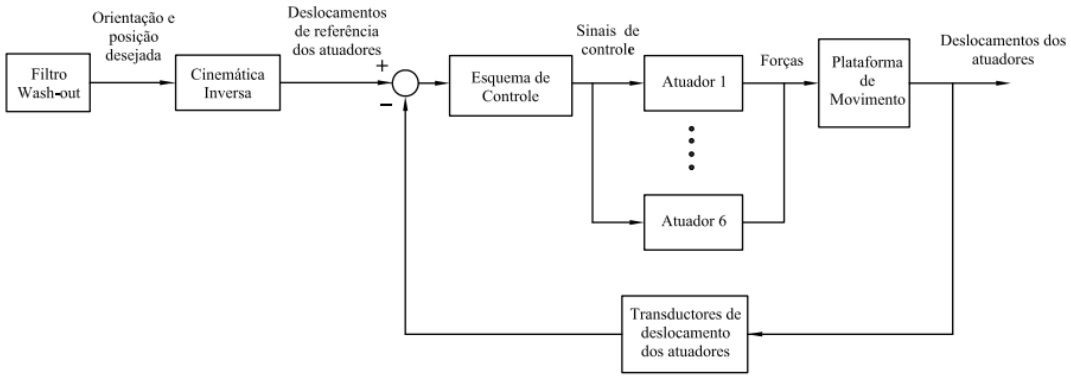


Figura 34 – Estrutura de controle no espaço das juntas proposta [3]

Se definiu que a principio se usaria a versão **descentralizada** desta estrutura, ou seja, o controle é dado em cada junta independentemente, e o acoplamento existente é tratado como uma perturbação. Sendo assim, se pode considerar o problema de controle equivalente ao de controlar seis sistemas SISO, o que facilita o projeta de controle.

5.1.2 Cinemática Inversa

Antes de realizar o projeto se deve projetar a resolução da cinemática inversa, necessária para esta estrutura. Isto se faz usando as equações 3.3 e embutindo elas em um programa que consiga resolvê-las.

Então se implemento o algoritmo 1 que, a partir do vetor de posição $\mathbf{t} = (x, y, z)$ e orientação $\mathbf{\Theta} = (\phi, \theta, \psi)$ em ângulos de Euler, calcula seu equivalente $\mathbf{L}$ no espaço das juntas. Admite-se que o algoritmo conhece os valores dos vetores $\mathbf{p}_i$ e $\mathbf{b}_i$.

Algorithm 1 Calculo da Cinemática Inversa

Require: Vetor de posição e orientação $(\mathbf{t}, \mathbf{\Theta})$
  Calcula $\mathfrak{R}(\mathbf{\Theta})$
  for $i = 0$ to 6 **do**
    $\mathbf{S}_i \leftarrow \mathfrak{R}\,\mathbf{p}_i + \mathbf{t}_i - \mathbf{b}_i$
    $L_i \leftarrow \|\mathbf{S}_i\|$
  end for
  return $\mathbf{L} \leftarrow \begin{bmatrix} L_1 & L_2 & \cdots & L_6 \end{bmatrix}^T$

O calculo da matriz de rotação se dá pela seguinte equação

$$\mathfrak{R}(\mathbf{\Theta}) = \begin{bmatrix} C\psi C\theta & C\psi S\theta S\phi - C\phi S\psi & C\psi C\phi S\theta + S\psi S\phi \\ C\theta C\psi & C\psi C\phi + S\psi S\theta S\phi & C\phi S\psi S\theta - C\psi S\phi \\ -S\theta & C\theta S\phi & C\theta C\phi \end{bmatrix}. \tag{5.1}$$

5.2 Identificação da Planta

Se definiu que se usaria uma estrutura descentralizada, isto quer dizer que se dividirá o problema de controle da plataforma em seis projetos de controladores no espaço das juntas. Assim é necessário identificar o comportamento das seis juntas ativas que o robô possui.

Aplicando a mesma lógica para todas as juntas, se pode fazer uma aproximação linear para um modelo de segunda ordem de cada junta. Estes modelos poderão ser usados para determinar o controlador que satisfaça os requisitos do problema.

Como discutido na seção 4.2, não é possível usar o método usual de identificação pela resposta rampa, uma vez que a planta é integradora. Ainda não é possível fazer uma identificação ainda que se realize uma derivada do sinal dos potenciômetros (velocidade), devido a que o sinal fica muito ruidoso para poder identificar os parâmetros essenciais como ganho e tempo de resposta, como se pode observar a figura 35.

A primeira vista, os sinais nem parecem seguir um degrau, embora os sinais originais tinham uma comportamento de rampa, como se pode observar na figura 36.

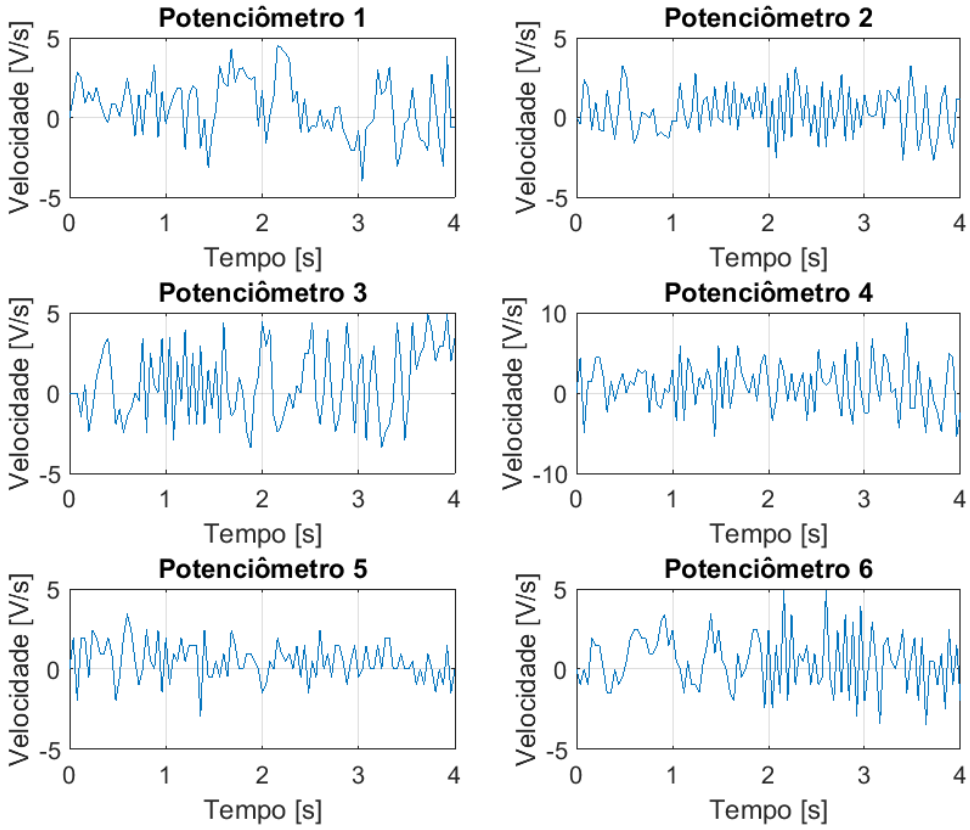


Figura 35 – Derivadas dos sinais dos transdutores em malha aberta para uma entrada degrau

Não seria correto realizar uma identificação usando um sinal filtrado, devido a que a frequência de corte necessária para filtrar o ruído se encontra muito perto da frequência natural da plataforma, isto é, os polos do filtro influenciam os polos da planta, efetivamente modificando a dinâmica da planta. Isto será demostrado na seção subsequente.

Os sinais dos potenciômetros (posição), quando o sistema se encontra em malha aberta não são muito ruidosos, embora as derivadas sejam. Por esta razão, se optou por teorizar uma forma de identificação usando a resposta rampa deste sistema.

Como foi discutido anteriormente, primeiramente se realiza uma aproximação da assintota que o sistema está seguindo. O MATLAB possui uma função denominada `polyfit`, que se encarrega de encontrar o melhor polinômio de ordem desejada que melhor se ajusta ao um determinado sinal. A assintota em questão é basicamente determinada por um polinômio de 1ª Ordem. O uso desta função facilita a identificação das assintotas quando se tem vários sistemas por identificar e com vários dados disponíveis.

A função `polyfit` retorna os coeficientes do melhor polinômio, que neste caso será de 1ª Ordem. Sabendo o ganho dos degraus que foram aplicados, se tem todas as informações para aplicar a teoria da seção 4.2.1.

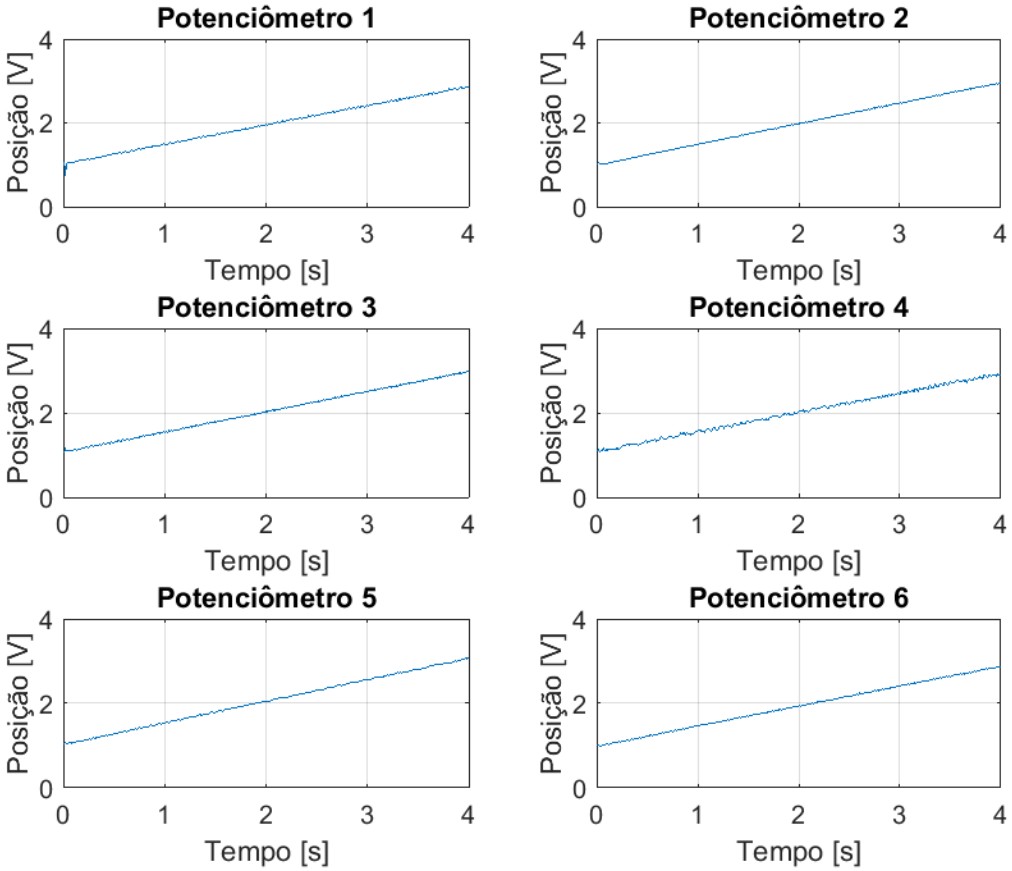


Figura 36 – Sinais dos transdutores em malha aberta para uma entrada degrau

As funções de transferência identificadas por este método são as seguintes

$$
\begin{aligned}
G_{p_1} &= \frac{U_1(s)}{Y_1(s)} = \frac{817.03}{s(s+34.39)^2}, \\
G_{p_2} &= \frac{U_2(s)}{Y_2(s)} = \frac{1076}{s(s+41.17)^2}, \\
G_{p_3} &= \frac{U_3(s)}{Y_3(s)} = \frac{5576.1}{s(s+91.68)^2}, \\
G_{p_4} &= \frac{U_4(s)}{Y_4(s)} = \frac{3355.7}{s(s+71.94)^2}, \\
G_{p_5} &= \frac{U_5(s)}{Y_5(s)} = \frac{756.42}{s(s+31.8)^2}, \\
G_{p_6} &= \frac{U_6(s)}{Y_6(s)} = \frac{1255.3}{s(s+43.08)^2}.
\end{aligned}
\tag{5.2}
$$

Elas modelam a resposta do potenciômetro, em V, de cada uma das pernas, para sua respectiva entrada de tensão no motor (sinal digital enviado pela DS1104).

5.2.1 Validação das Identificações

Para validar estos modelos foram realizados testes no Simulink com dados gravados e comparando-os a resposta da sua respectiva função de transferência. Estas comparações se podem observar nas figuras 37, 38, 39, 40, 41 e 42.

Com este tipo de comparação se pode verificar que o erro do ganho modelado, com o ganho real do sistema é mínimo, mas é difícil apreciar se os polos modelados são corretos devido ao polo integrador da planta.

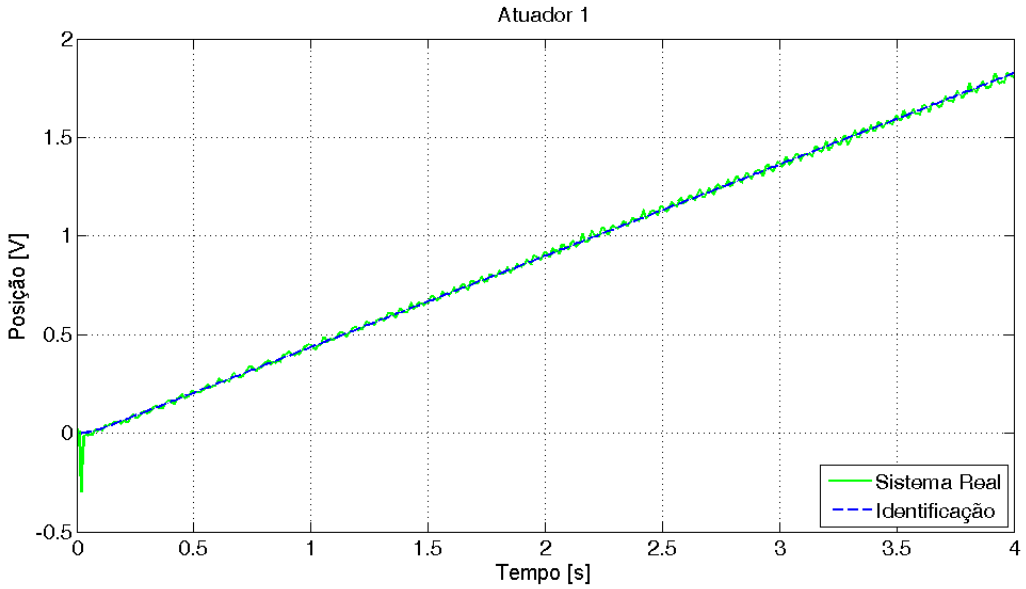


Figura 37 – Comparação da resposta a um degrau, do modelo do Atuador 1, com dados reais de uma resposta ao mesmo degrau no mesmo atuador

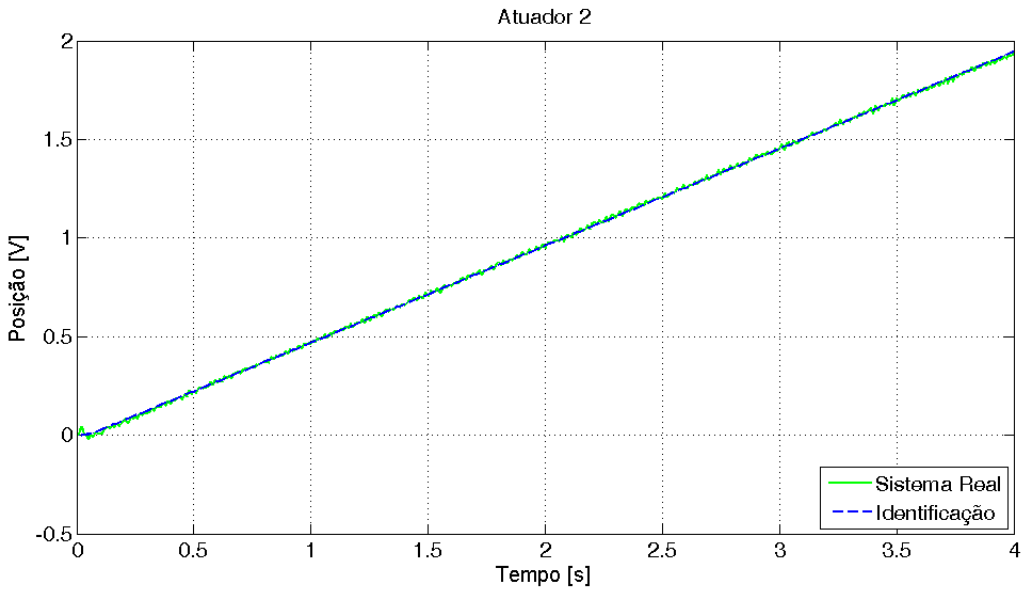


Figura 38 – Comparação da resposta a um degrau, do modelo do Atuador 2, com dados reais de uma resposta ao mesmo degrau no mesmo atuador

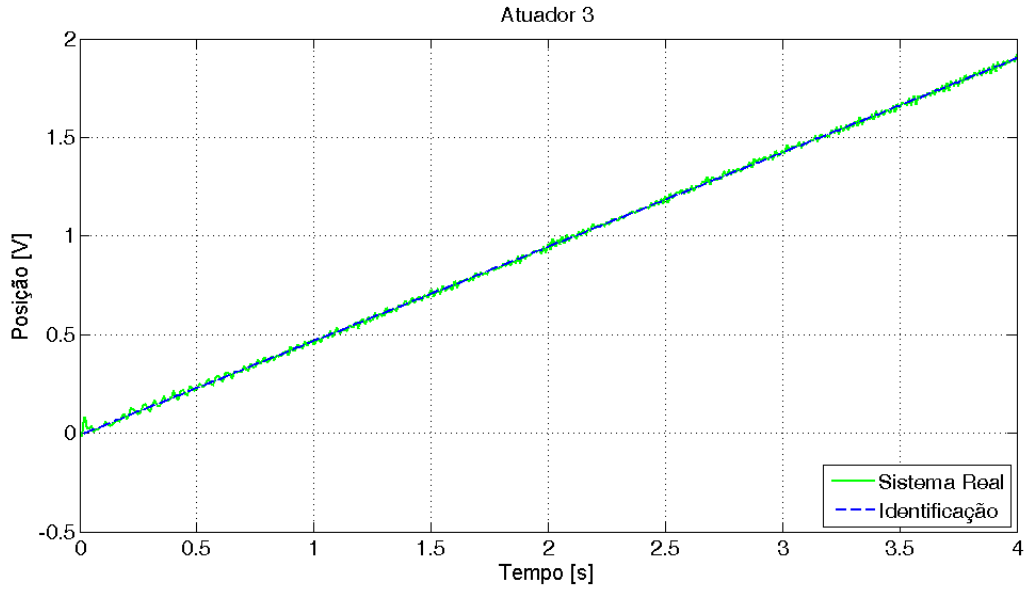


Figura 39 – Comparação da resposta a um degrau, do modelo do Atuador 3, com dados reais de uma resposta ao mesmo degrau no mesmo atuador

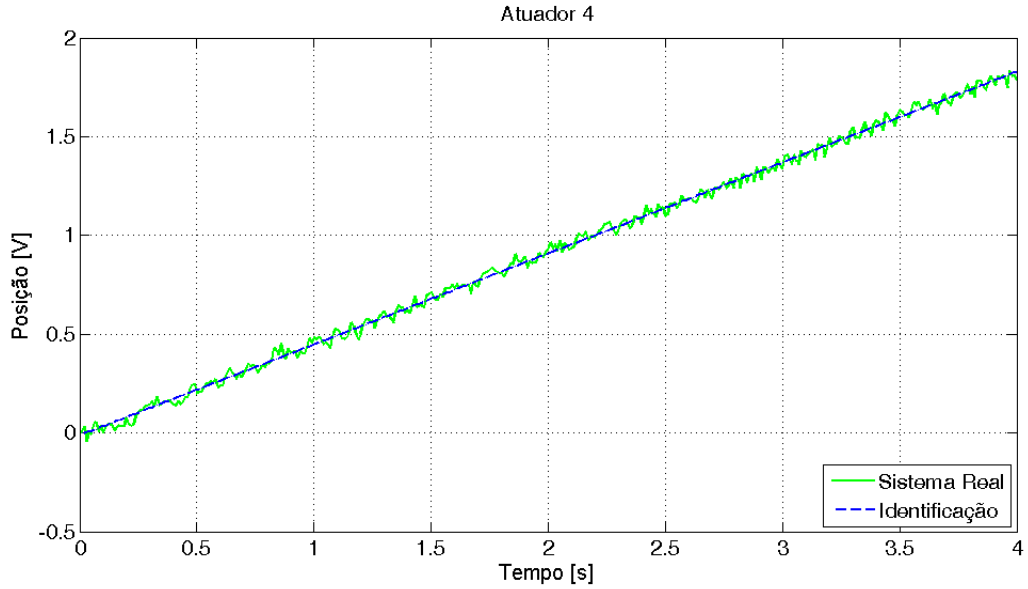


Figura 40 – Comparação da resposta a um degrau, do modelo do Atuador 4, com dados reais de uma resposta ao mesmo degrau no mesmo atuador

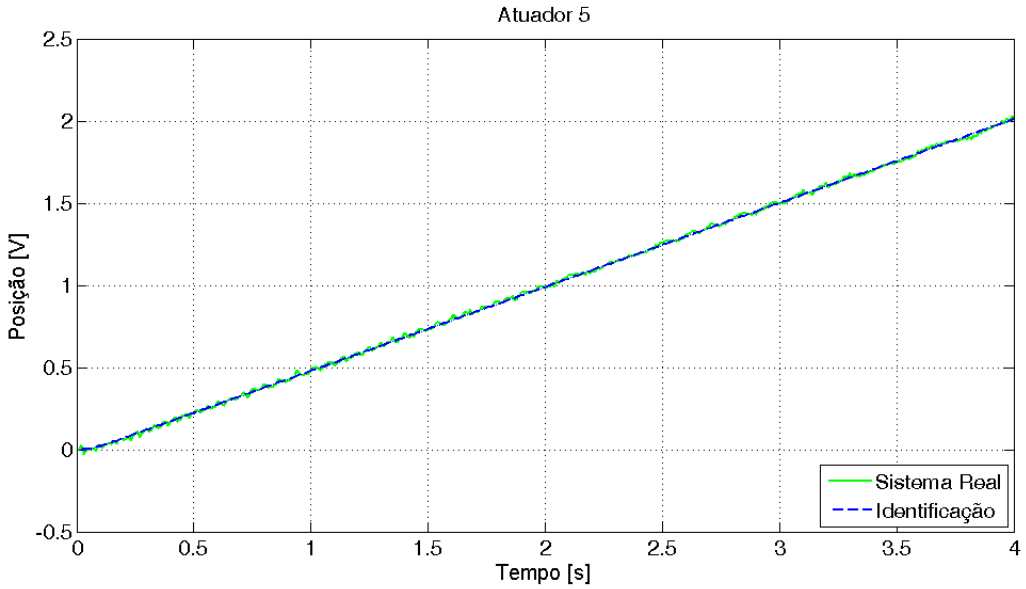


Figura 41 – Comparação da resposta a um degrau, do modelo do Atuador 5, com dados reais de uma resposta ao mesmo degrau no mesmo atuador

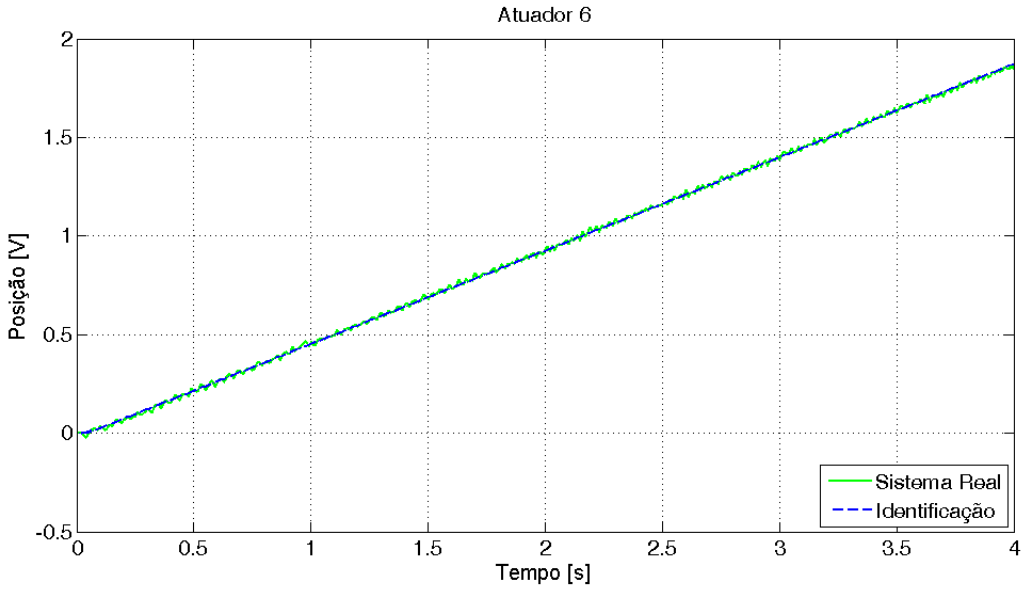


Figura 42 – Comparação da resposta a um degrau, do modelo do Atuador 6, com dados reais de uma resposta ao mesmo degrau no mesmo atuador

Para verificar se os polos não integradores modelados tem uma dinâmica condizente com a realidade se precisaria retirar o polo integrador, em outras palavras, derivar ambos sinais, contudo, como se viu na figura 35, no sinal original derivado não se pode detectar uma dinâmica de resposta a um degrau.

Para contornar este problema se usou um filtro de 1ª Ordem, tanto nos dados gravados da planta, quanto nas funções identificadas. Ao usar o filtro nos dois sinais se evita que a dinâmica do filtro distorça a comparação, assim se pode realizar uma

comparação correta entre os sinais e validar que o regime transitório do modelo se aproxime satisfatoriamente à realidade.

Os resultados desta comparação se podem observar nas figuras 43, 44, 45, 46, 47 e 48. Nelas se pode apreciar que as dinâmicas da modelagem tem um comportamento muito parecido com os dados reais.

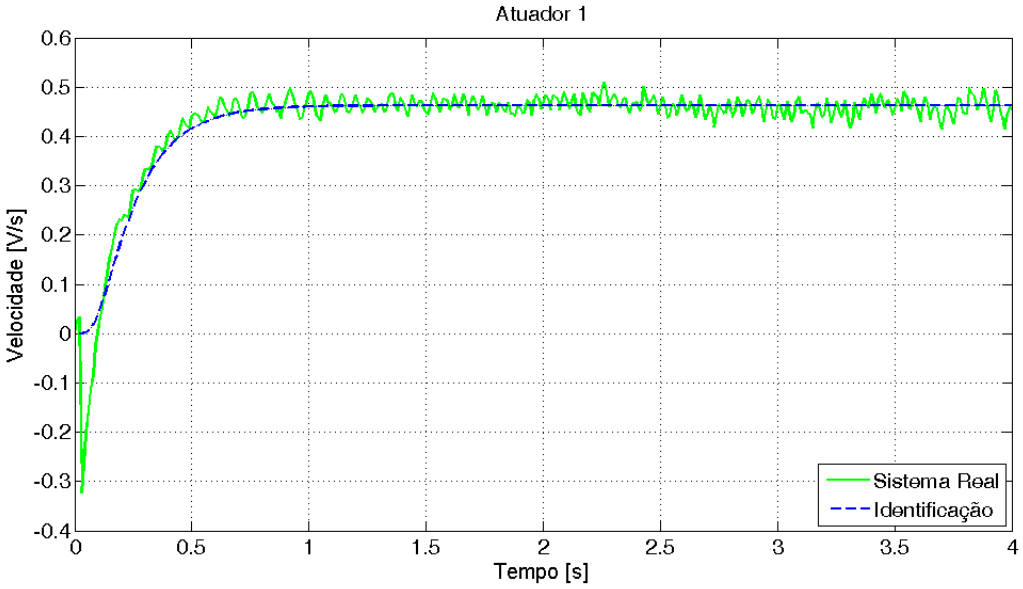


Figura 43 – Comparação da resposta a um degrau filtrada, do modelo do Atuador 1 sem o polo integrador, com dados reais filtrados e derivados de uma resposta ao mesmo degrau no mesmo atuador

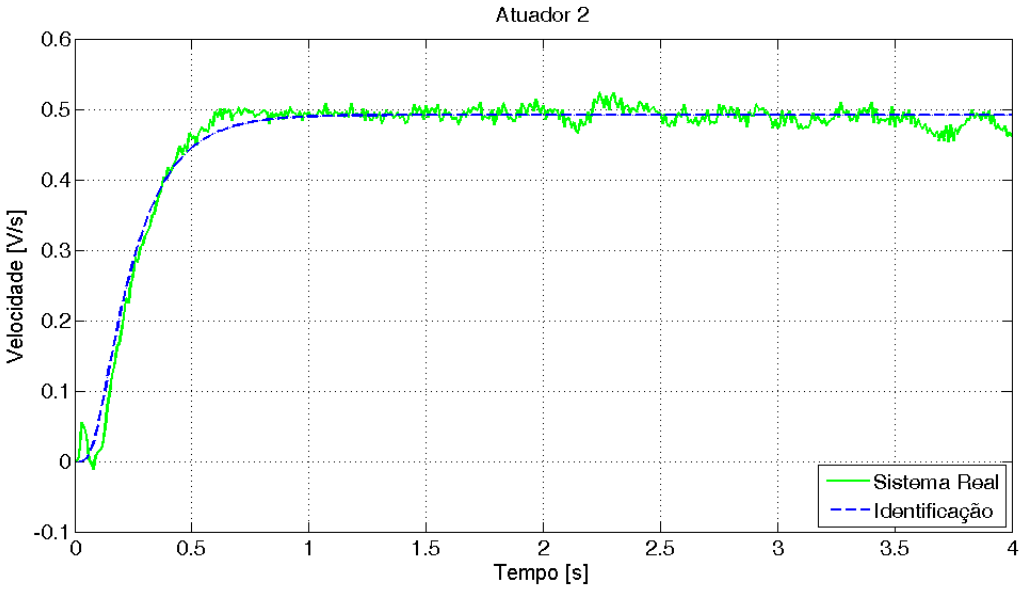


Figura 44 – Comparação da resposta a um degrau filtrada, do modelo do Atuador 2 sem o polo integrador, com dados reais filtrados e derivados de uma resposta ao mesmo degrau no mesmo atuador

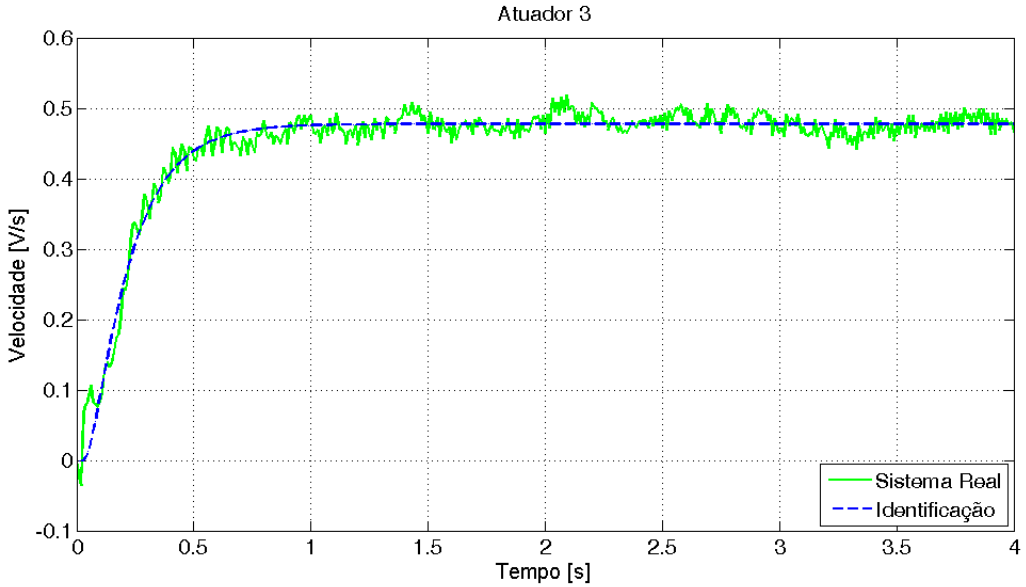


Figura 45 – Comparação da resposta a um degrau filtrada, do modelo do Atuador 3 sem o polo integrador, com dados reais filtrados e derivados de uma resposta ao mesmo degrau no mesmo atuador

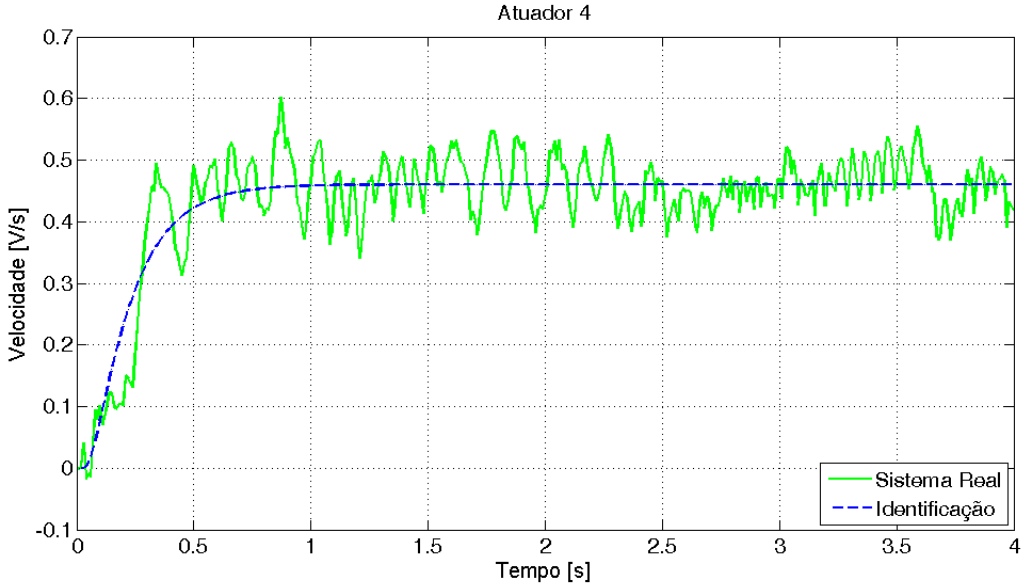


Figura 46 – Comparação da resposta a um degrau filtrada, do modelo do Atuador 4 sem o polo integrador, com dados reais filtrados e derivados de uma resposta ao mesmo degrau no mesmo atuador

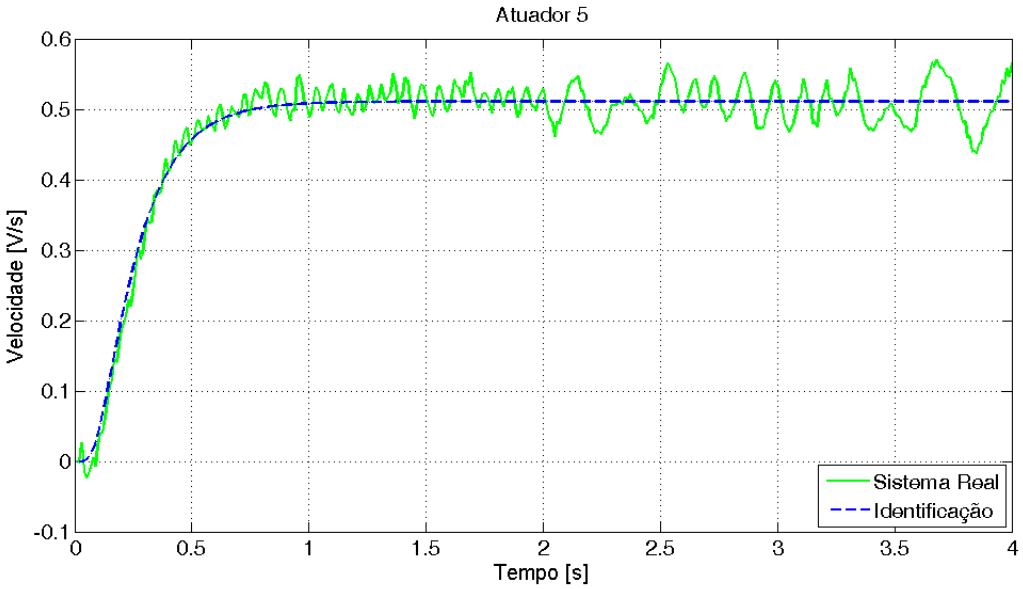


Figura 47 – Comparação da resposta a um degrau filtrada, do modelo do Atuador 5 sem o polo integrador, com dados reais filtrados e derivados de uma resposta ao mesmo degrau no mesmo atuador

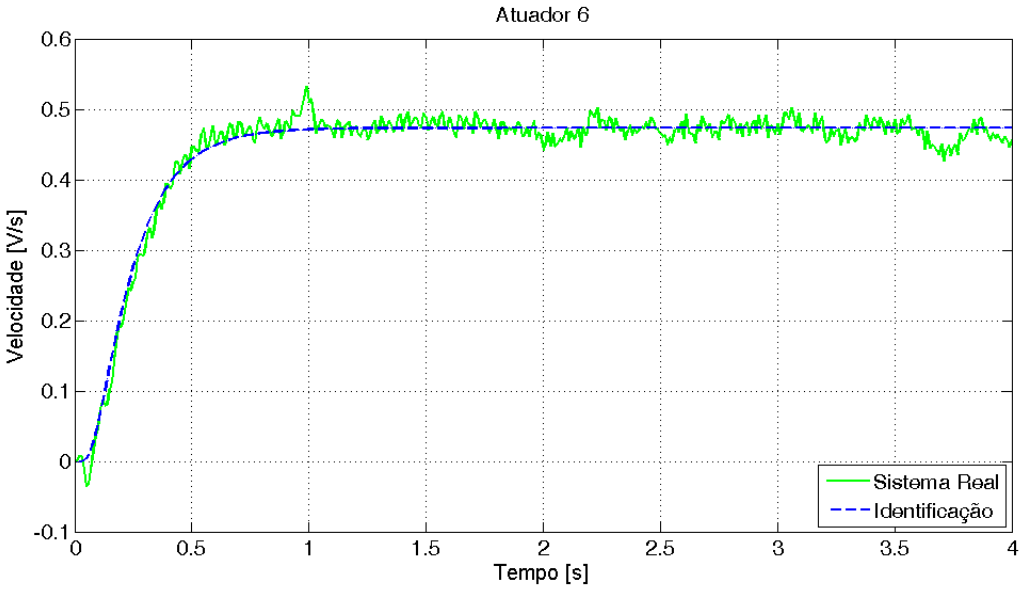


Figura 48 – Comparação da resposta a um degrau filtrada, do modelo do Atuador 6 sem o polo integrador, com dados reais filtrados e derivados de uma resposta ao mesmo degrau no mesmo atuador

Com base nestas comparações se pode concluir que os modelos identificados usando a metodologia desenvolvida são condizentes com a realidade, assim validando-se as identificações realizadas. Com estes modelos é possível realizar o projeto dos controladores.

5.2.2 Modelo no Espaço de Estados

Como visto na seção 3.4.2.1 para poder usar as técnicas de controle moderno é indispensável ter um modelo no espaço de estados, e usando o método apresentado na seção 4.3 se pode realizar a transformação dos modelos da planta (equação 5.2)

Neste caso o modelo teria 3 estados, não obstante, caso se adicione filtros de medida, o modelo a ser usado terá 3, mais a ordem do filtro, estados. Isto quer dizer que, se o filtro tem uma ordem elevada, os modelos no espaço de estados terão também mais estados. O que pode ser não benéfico, do ponto de vista do observador, pois este deverá observar mais estados e como cada estado é uma derivada do anterior o ruído é amplificado.

Isto se deve a que a função de transferência total do sistema passa a ser

$$G_T(s) = G_p(s)F(s), \tag{5.3}$$

onde os $G_p(s)$ é o modelo para a posição e $F(s)$ é o filtro de medida usado.

Ressalta-se que em sistemas integrativos, sua representação no espaço de estados, a primeira coluna da matriz $\mathbf{A}$ é conformada por zeros, ou seja, na equação 4.28 o elemento a_n é sempre igual a zero.

5.3 Filtros de medida Projetados

Sabendo-se qual a estrutura do sistema de controle e determinando-se um modelo da planta se pode começar o projeto de controle. Não obstante, devido à natureza do sinal usado para realimentar e fechar a malha de controle, é necessário projetar antes um filtro para retirar o ruído.

O filtro a ser projeto deverá ser tal que consiga, filtrar satisfatoriamente o ruído presente e que respeite a frequência na qual se quer operar a plataforma. Com as informações mostradas na seção 3.1 e os conceitos discutidos na seção 4.1, se pode deduzir que o filtro a ser projetado é um filtro passa-baixa com uma frequência de corte de no mínimo 2 ou 3 Hz.

Outra característica desejada do filtro é que sua presença não atrase muito o sinal do transdutor, pois teria que se usar ganhos maiores no controle para compensar pelo atraso e poder manter o compromisso com o tempo de resposta desejado do Sistema de Movimento.

Antes de poder projetar o filtro se tem que analisar o espectro de frequência do sinal da medida de modo a determinar em quais bandas de frequência o ruído é mais prominente.

5.3.1 Análise Espectral

Se realizaram testes na plataforma onde se exitava as juntas com senoides de uma determinada frequência e se gravava os dados, para poder analisar posteriormente.

Para não forçar muito os motores das juntas, todas senoides dos testes feitos são aplicadas ao mesmo tempo em cada junta, e os valores de frequência e amplitude são iguais para cada junta. Devido à cinemática da plataforma, se pode considerar que o movimento das juntas ao mesmo tempo evita que aconteça acoplamento entre os movimentos das juntas.

Os testes foram com uma frequência baixa ($\omega = 2\,\text{rad/s}$), uma média ($\omega = 4\,\text{rad/s}$) e outra alta $\omega = 6\text{rad/s}$. Devido a limitações físicas na velocidade dos motores, o valor da amplitude das senoides também é alterado conforme se altera a frequência usando a seguinte equação

$$A_{\max} = \frac{v_{\max}}{\omega}. \tag{5.4}$$

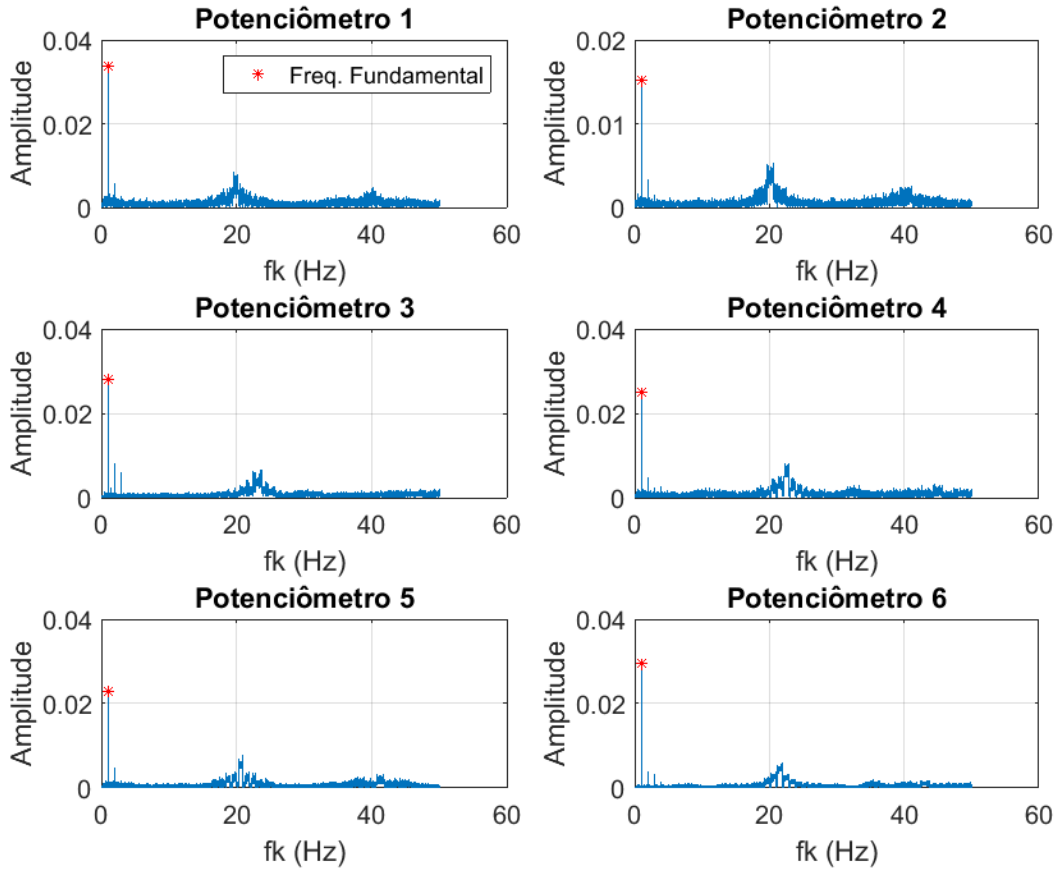


Figura 49 – Análise espectral dos sinais dos potenciômetros para uma entrada senoidal de frequência 6 rad/s em cada junta ativa em malha aberta

Sabendo-se experimentalmente o valor de $v_{\max}$ se pode determinar qual a amplitude máxima $A_{\max}$ que se pode usar como referência para garantir que o atuador funcione sem ultrapassar seus limites, isto é, que trabalhe numa região linear.

O resultado da análise espectral feita no MATLAB, para uma das frequências pode ser observada na figura 49. A partir da análise se nota que existem basicamente duas bandas onde o ruído prevalece em todas as juntas, numa região perto dos 20 Hz e outra perto da própria frequência da senoide de referência.

Esta característica se repetiu ao fazer a análise para as outras frequências, ou seja, existia ruído concentrado perto da própria frequência com que se excitava os atuadores, como se pode observar na figura 50.

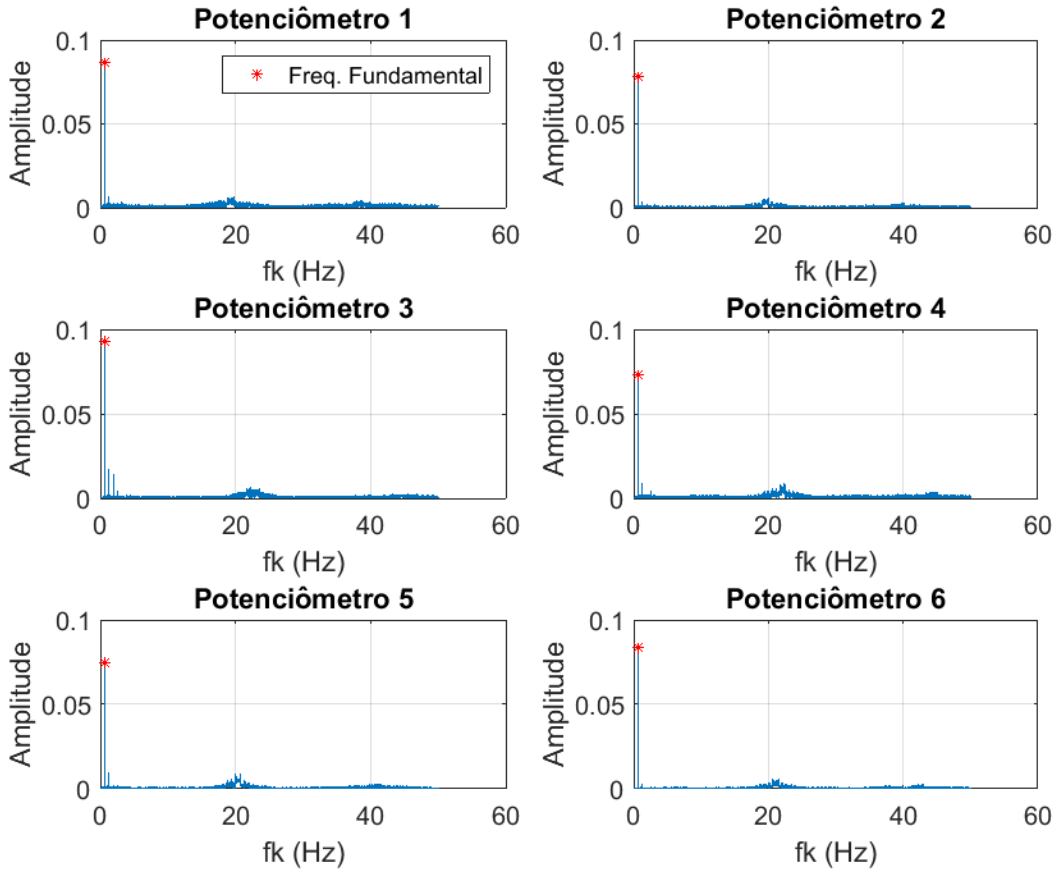


Figura 50 – Análise espectral dos sinais dos potenciômetros para uma entrada senoidal de frequência 4 rad/s em cada junta ativa em malha aberta

Se pode concluir que a frequência de corte dos filtros deve ser a máxima, antes de influenciar o desempenho do controlador, ou seja a frequência de corte dever ser 3 Hz. Com isto se pode garantir que o controlador ainda consiga cumprir os requisitos descritos na seção 3.1.

No entanto, testes preliminares, no Simulink, com frequências de corte de 2 ou 3 Hz mostraram que os filtros eram ineficientes para filtrar o ruído. Devido a estas razões, se optou por utilizar como frequência de corte perto de 1 Hz no projeto dos filtros.

5.3.2 Filtro de 1ª Ordem

Este é o filtro mais simples projetado, onde se usa apenas a frequência de corte, para posicionar o polo do filtro. Como se viu na seção 4.1, o filtro de 1ª ordem passa-baixa mais simples é descrito pela equação 4.2.

Se sabe que a frequência de corte de um filtro de 1ª ordem é igual ao inverso da sua constante de tempo, assim a seguinte função de transferência representa o filtro que se quer

$$H(s) = \frac{1}{\frac{1}{5.815}s + 1} = \frac{5.815}{s + 5.815} \tag{5.5}$$

A simplicidade deste filtro nos garante que o atraso que causa no sinal medido será mínimo. Este filtro é mais simples de ser considerado ao realizar o projeto do controle, pois se adiciona apenas um polo ao sistema equivalente. Isto se aplica tanto para as técnicas de controle clássicas quanto às modernas.

A figura 51, mostra a resposta a magnitude da resposta na frequência da equação 5.5. Se nota que a frequência de corte se encontra perto de 1 Hz.

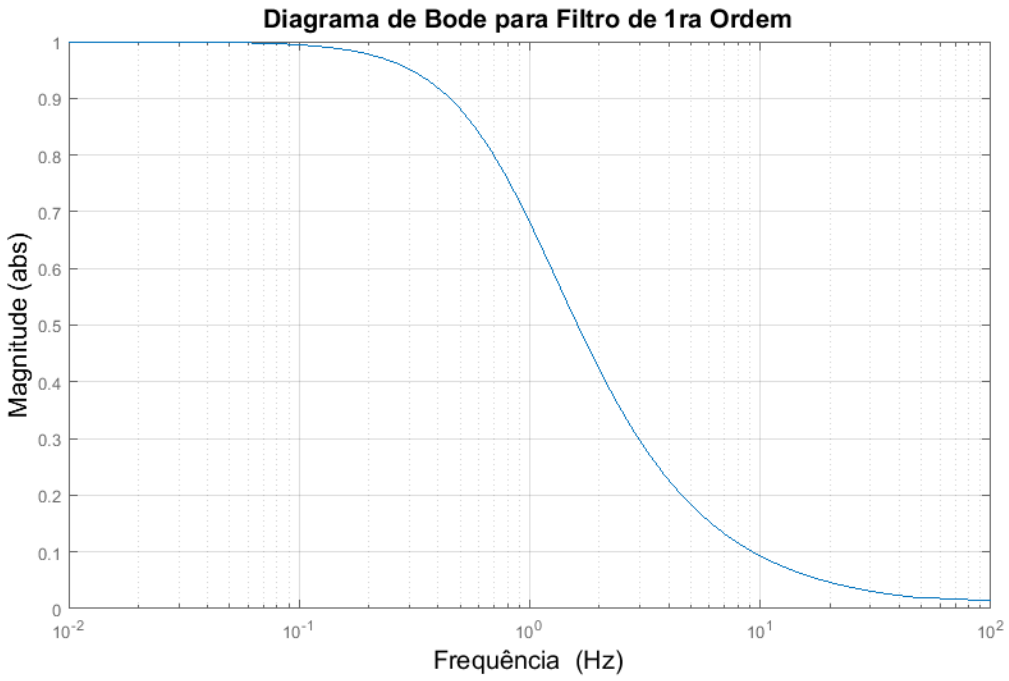


Figura 51 – Magnitude da resposta na frequência do filtro de 1ª Ordem

Para testar a eficácia do filtro projetado se realizaram vários testes na plataforma com diversas senoides de entrada em malha aberta e se gravaram os dados, similar aos testes feitos para realizar a analise espectral do ruído na medida. Posteriormente estes dados gravado foram usados como entrada pela função 5.5, e se analisaram os resultados.

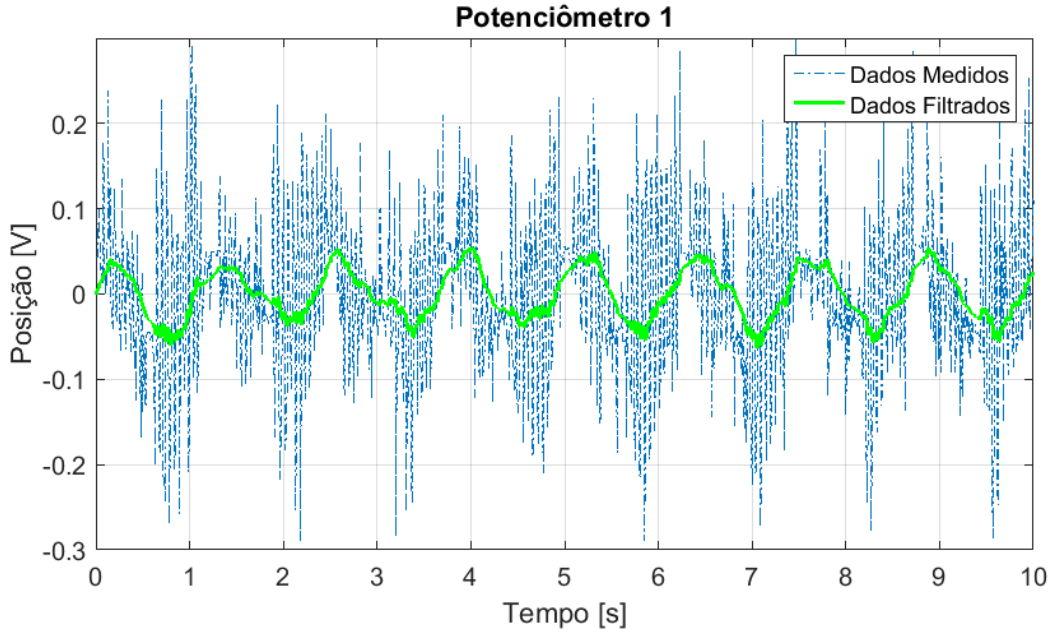


Figura 52 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 5 rad/s, com elas filtradas pelo filtro de 1ª Ordem

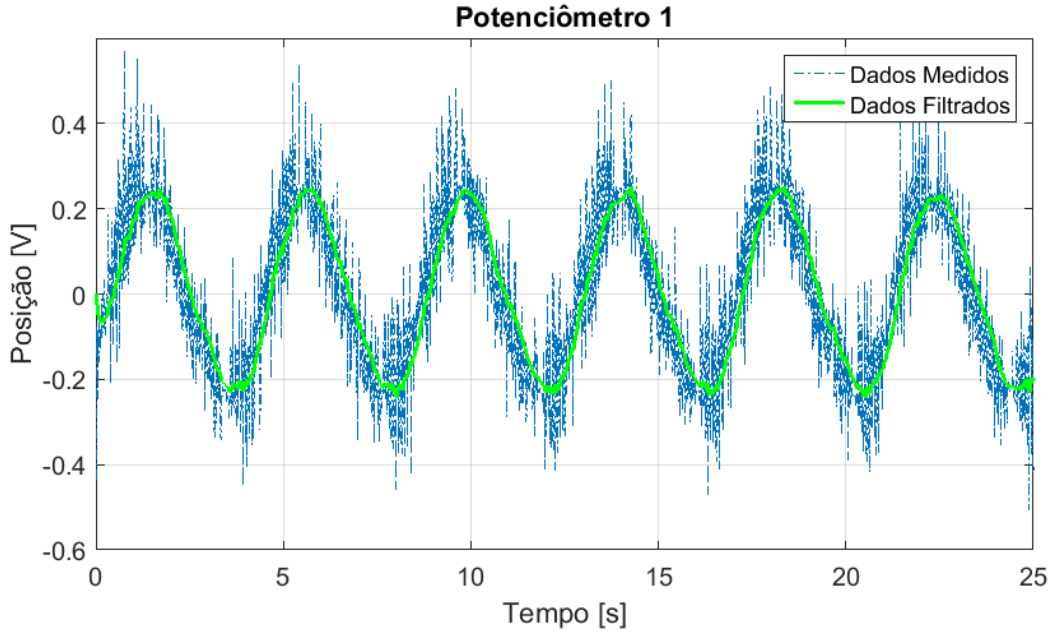


Figura 53 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 1.5 rad/s, com elas filtradas pelo filtro de 1ª Ordem

Como se tem 6 potenciômetros para analisar e foram usadas 6 senoides diferentes nos testes, não é prático mostrar todos os resultados neste documento, portanto se apresentará

os resultados para apenas uma das leituras com duas frequências de entrada diferentes. Estes podem ser observados nas figuras 53 e 52.

Como se podia apreciar pela diferença das figuras 50 e 49, se destaca a dinâmica do ruído presente na leitura, que depende da frequência da senoide de entrada da respectiva junta. Isto que dizer, que quanto maior for a frequência da senoide, pior será a ruído. Isto se deve a que, como visto na seção 5.3.1, quanto maior a frequência da senoide, menor deverá ser a amplitude desta, então o ruído será mais perceptível em amplitudes menores, enquanto que amplitudes maiores, e por consequência, frequências menores, a influencia do ruído diminui.

Se pode concluir que este filtro causa um atraso de menos de 0.1 s, o que é relativamente pequeno. Possui bom desempenho em termos de filtragem do ruído e média distorção do sinal, para baixas frequências. Enquanto que para altas frequências, possui médio desempenho, devido a que existem distorções notáveis. Os resultados para os outros potenciômetros e para as outras frequências segue este mesmo padrão.

5.3.3 Filtro Butterworth

O filtro de 1ª Ordem, é considerado um filtro do tipo Butterworth, no entanto a partir desta seção os filtros foram projetado usando comandos do MATLAB. Neste caso se usa as funções `buttord` e `butter`.

A função `buttord(Wp,Ws,Rp,Rs)` nos permite calcular a ordem miníma de um filtro Butterworth e sua frequência de corte correspondente que atenda um conjunto de requisitos. Estes requisitos são as frequências de canto tanto para a banda passante (`Wp`) como para banda rejeitada (`Ws`) em rad/s. Assim como o valor máximo de *ripple* que se admite na banda passante (`Rp`) e a atenuação miníma desejada na banda rejeitada (`Rs`), análogo ao *ripple* na banda rejeitada, ambos em dB. [21]

Já, a função `butter(n,Wn)` calcula os coeficientes da função de transferência de um filtro Butterworth de ordem `n`, com frequência de corte `Wn`. Portanto se usa a função `buttord` para determinar os parâmetros necessários para usar função `butter`. [22]

A princípio o único requisito que se tem é a frequência de corte, com base nisso se decidiu usar um `Wp` de 0.95 Hz. Os parâmetros usados no projeto inicial se encontram na tabela 1.

`Rp` [abs]	`Rs` [abs]	`Wp` [Hz]	`Ws` [Hz]
0.99	0.01	0.95	5

Tabela 1 – Requisitos usados para o projeto inicial do filtro Butterworth

Estes valores foram convertidos para as unidades necessárias do função `buttord`,

para o caso da magnitude se transformou para dB usando a equação a seguir.

$$x_{\text{dB}} = -20\log_{10}(x_{\text{abs}}) \tag{5.6}$$

Para as frequências é necessário fazer a conversão para rad/s basta multiplicar por 2π, assim se pode usar a formula a seguir para fazer a conversão.

$$f_{\text{rad/s}} = 2\pi f_{\text{Hz}} \tag{5.7}$$

Usando os dados da tabela 1 se consegue calcular a ordem miníma do filtro `n` com uma frequência de corte `Wn`. O resultado calculado pelo MATLAB foi ordem igual a 4 e frequência de corte de aproximadamente 1.58 Hz. Com estes dados se calcula a função de transferência usando a outra função `butter`.

A função de transferência do calculado pelo MATLAB é a seguinte

$$F_B(s) = \frac{Y_f(s)}{Y(s)} = \frac{9741}{s^4 + 25.96s^3 + 337s^2 + 2562s + 9741}. \tag{5.8}$$

Se ressalta a simplicidade do filtro Butterworth, pois ele não possui zeros na sua função de transferência. A magnitude da resposta na frequência desta função pode ser observada na figura 54 usando os requisitos da tabela 1.

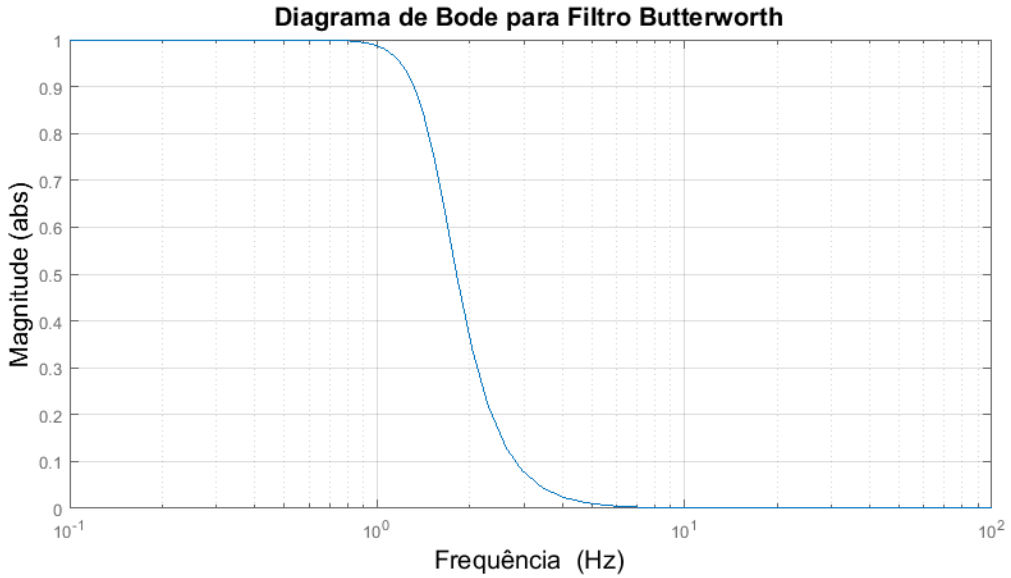


Figura 54 – Magnitude da resposta na frequência do filtro Butterworth, projetado com a tabela 1

Igualmente ao filtro de 1ª Ordem, se usaram os mesmos dados gravados para fazer um comparação entre o sinal original e sinal filtrado pela função 5.8.

Da mesma forma, serão apresentados apenas duas comparações para senoides de diferente frequências, para um mesmo transdutor. Isto pode ser observado nas figuras 55 e 56, onde se usou o mesmo transdutor dos resultados usados para o Filtro de 1ª Ordem.

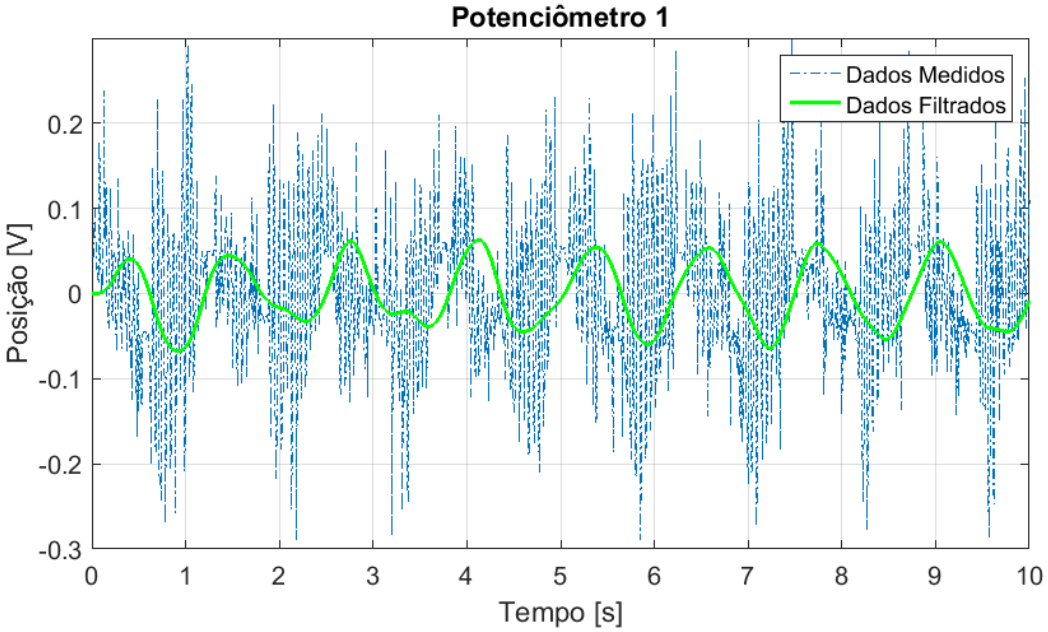


Figura 55 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 5 rad/s, com elas filtradas pelo filtro Butterworth

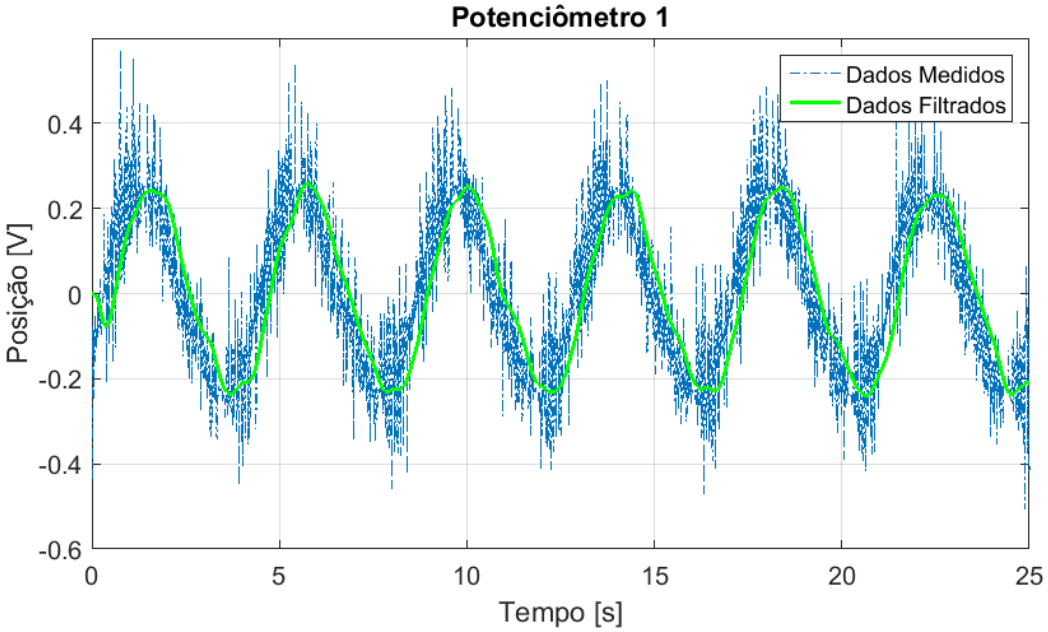


Figura 56 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 1.5 rad/s, com elas filtradas pelo filtro Butterworth

Conclui-se que o filtro Butterworth projetado, tem um atraso menor a 0.15 s, maior que o filtro de 1ª Ordem. Em baixas frequências tem um ótimo desempenho para filtrar o ruído, e pouca distorção. O mesmo se aplica no caso de ter uma frequência alta. Igualmente este resultado é similar para os outros potenciômetros.

Os parâmetros usados na tabela 1, são os usados para o projeto preliminar, no entanto, se realizaram mais projetos ajustando estes parâmetros individualmente e analisando sua influência na filtragem do sinal medido. Contudo estas analises não serão apresentadas neste documento, pois, como será explicado na seção 5.4, se optou por usar o filtro de 1ª Ordem. Isto também se aplica para o resto dos filtros projetados.

5.3.4 Filtro Chebyshev Tipo II

Se optou por projetar o filtro Chebyshev do Tipo II, ao invés do tipo I, devido a que este possui não possui *ripple* na banda passante, e *ripple* na banda rejeitada não é tão crítico para o desempenho do filtro na malha de controle. Igualmente ao filtro Butterworth, se usou as funções do MATLAB, `cheb2ord` e `cheby2`.

Similar a `buttord`, `cheb2ord(Wp,Ws,Rp,Rs)` calcula a ordem miníma do filtro do Chebyshev Tipo II que atenda os requisitos de frequências de canto (`Wp` e `Ws`) e *ripple* (`Rp` e `Rs`). [23]

A função `cheby2(n,Rs,Ws)` calcula os coeficientes da função de transferência de um filtro Chebysehv Tipo II de ordem `n`, com frequência de canto de banda passante `Ws` e atenuação miníma de `Rs` decibéis na banda rejeitada. Da mesma forma se usa `cheb2ord` para determinar os parâmetros necessários para calcular a função de transferência do filtro com `cheby2`. [24]

Se nota que a função para calcular o filtro precisa de mais um parâmetro, em contrapartida da sua versão para o filtro Butterworth. Este é o mesmo que foi usado em `cheb2ord`. Os parâmetros usados no projeto inicial são os mesmos do filtro Butterworth (tabela 1). Igualmente se usa as equações 5.7 e 5.6, para transformar os valores e se aplicam as funções.

A ordem calculada é 4 e a frequência de corte é aproximadamente 1.48 Hz. Usando estes resultados e os requisitos a função de transferência calculada é

$$F_C(s) = \frac{Y_f(s)}{Y(s)} = \frac{0.01(s^2 + 412.1)(s^2 + 2402)}{(s^2 + 18.93s + 109.9)(s^2 + 6.42s + 90.04)}. \tag{5.9}$$

A magnitude da resposta na frequência desta função se ilustra na figura 57. Se pode observar a característica do *ripple* presente na banda rejeita, embora que ,devido aos parâmetros usados, ele seja pouco perceptível.

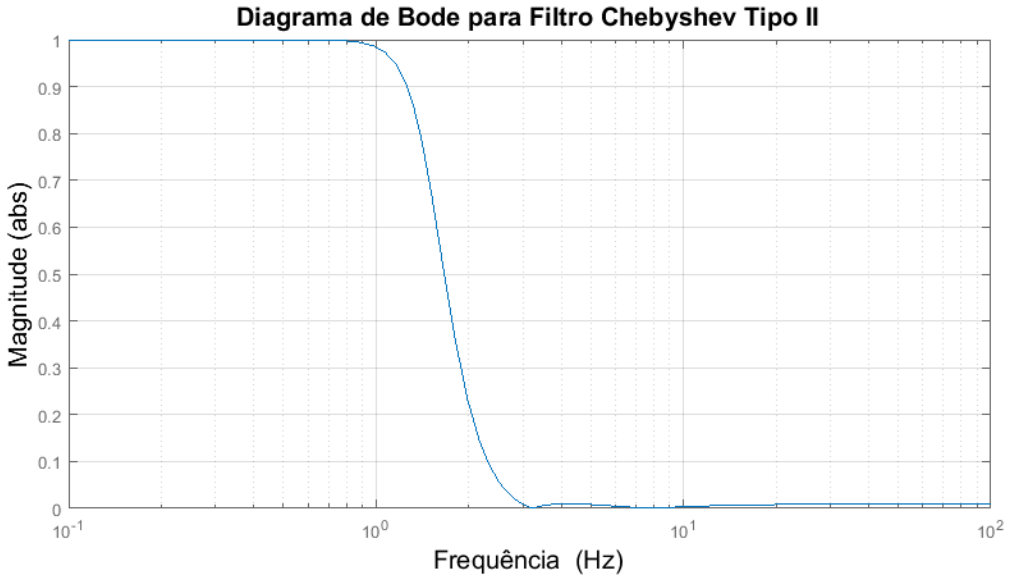


Figura 57 – Magnitude da resposta na frequência do filtro Chebyshev Tipo II, projetado com a tabela 1

Se usaram os mesmos dados gravados para fazer um comparação entre o sinal original e sinal filtrado pela função 5.9. Isto pode ser observado nas figuras 58 e 59.

Conclui-se que o filtro Chebyshev Tipo II projetado, tem um atraso de aproximadamente 0.15 s, marginalmente menor que o filtro Butterworth. Em baixas frequências tem um ótimo desempenho para filtrar o ruído, e pouca distorção. O mesmo se aplica no caso de ter uma frequência alta, exceto que a distorção é quase inexistente. Este resultado é similar para os outros potenciômetros.

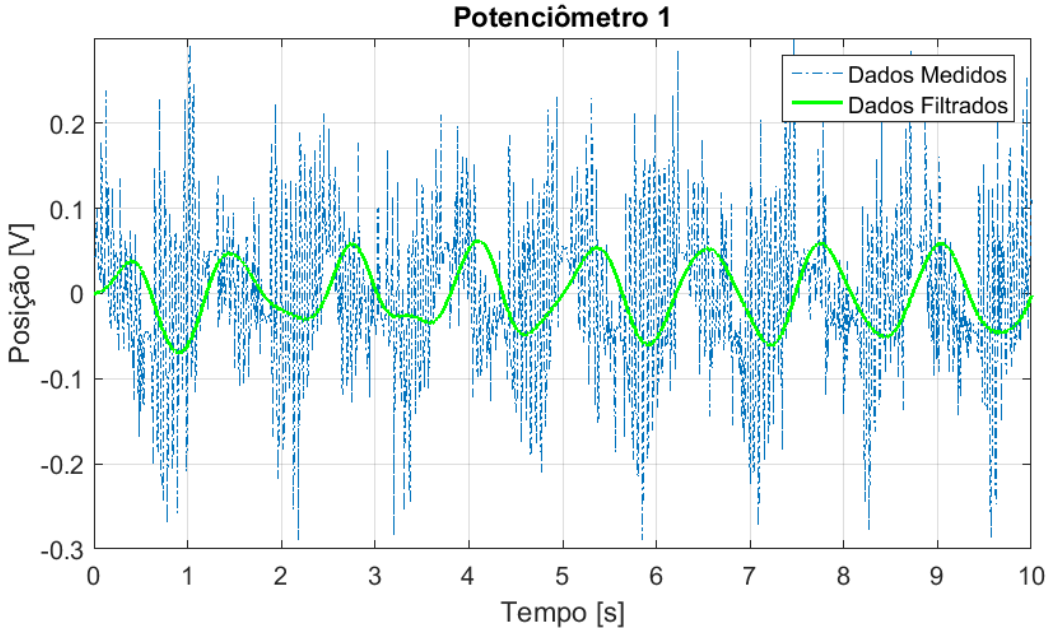


Figura 58 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 5 rad/s, com elas filtradas pelo filtro Chebyshev Tipo II

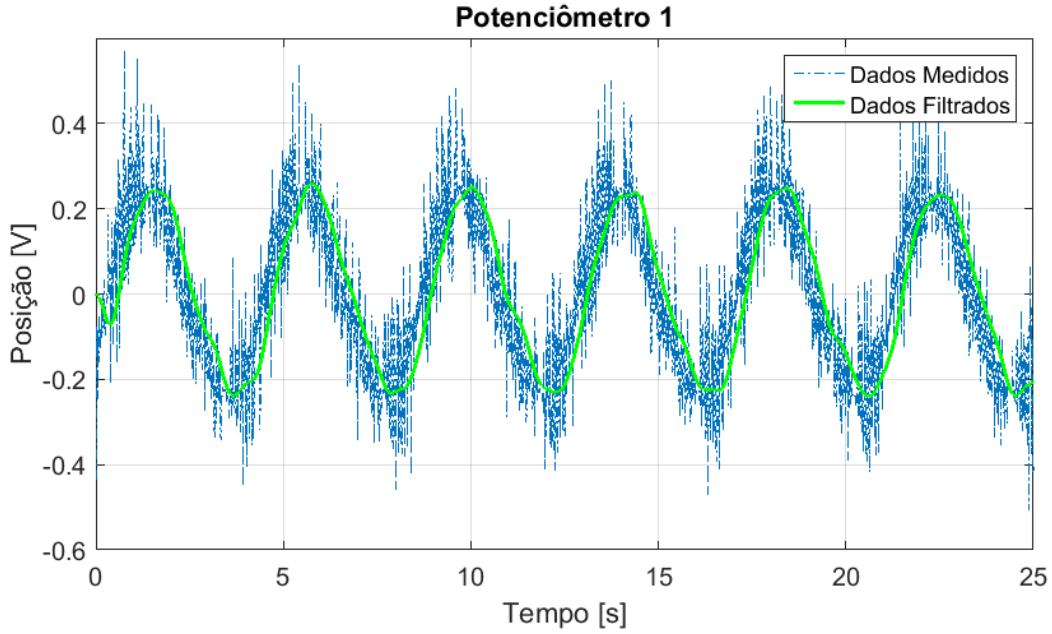


Figura 59 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 1.5 rad/s, com elas filtradas pelo filtro Chebyshev Tipo II

5.3.5 Filtro Elíptico

Se usou as funções do MATLAB, `ellipord` e `ellip`. Similar aos outros 2 casos, `ellipord(Wp,Ws,Rp,Rs)` calcula a ordem miníma do filtro do Elíptico. [25]

A função `ellip(n,Rp,Rs,Ws)` calcula os coeficientes da função de transferência de um filtro Elíptico de ordem `n`, com frequências de canto `Ws`, e *ripples* máximos permitidos `Rp` e `Rs`. [24]

Da mesma forma o projeto do filtro elíptico no MATLAB precisa usar mais parâmetros. Como o filtro Elíptico consegue ter a menor banda transitória de todos o filtro vistos na seção 4.1. Se usou frequências de canto diferentes para o projeto inicial deste filtro, todos os parâmetros usados são apresentados na tabela 2.

`Rp` [abs]	`Rs` [abs]	`Wp` [Hz]	`Ws` [Hz]
0.99	0.01	1	3

Tabela 2 – Requisitos usados para o projeto inicial do filtro Elíptico

A ordem calculada é 3 e a frequência de corte é aproximadamente 1.31 Hz. Usando estes resultados e os dados da tabela 2 a função de transferência calculada é

$$F_E(s) = \frac{Y_f(s)}{Y(s)} = \frac{0.63(s^2 + 608.8)}{(s + 6.22)(s^2 + 5.59s + 61.78)} \tag{5.10}$$

A magnitude da resposta na frequência desta função se ilustra na figura 57.

Se pode observar a característica do *ripple* presente na banda rejeita e na banda passante como se exemplificou na figura 28, embora devido aos parâmetros usados ele seja pouco perceptível.

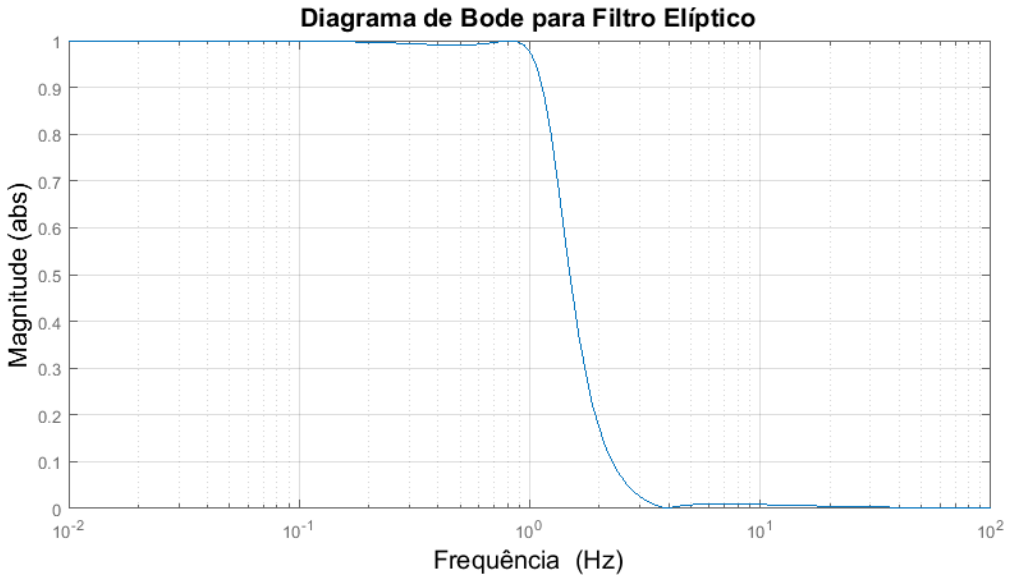


Figura 60 – Magnitude da resposta na frequência do filtro Elíptico, projetado com a tabela 2

Se usaram os mesmos dados gravados para fazer um comparação entre o sinal original e sinal filtrado pela função 5.10. Isto pode ser observado nas figuras 61 e 59.

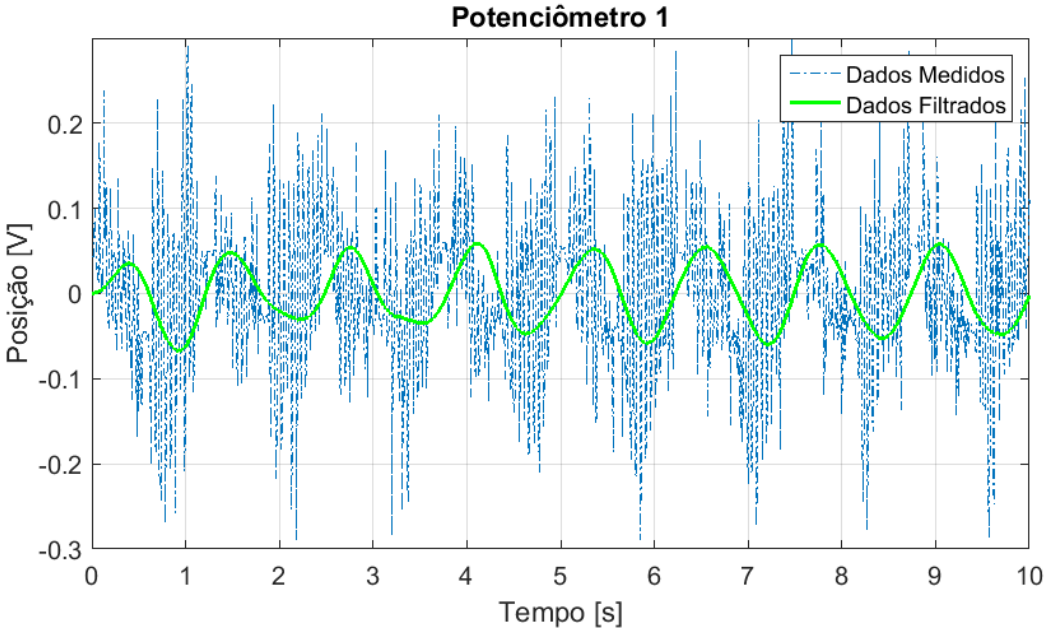


Figura 61 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 5 rad/s, com elas filtradas pelo filtro Elíptico

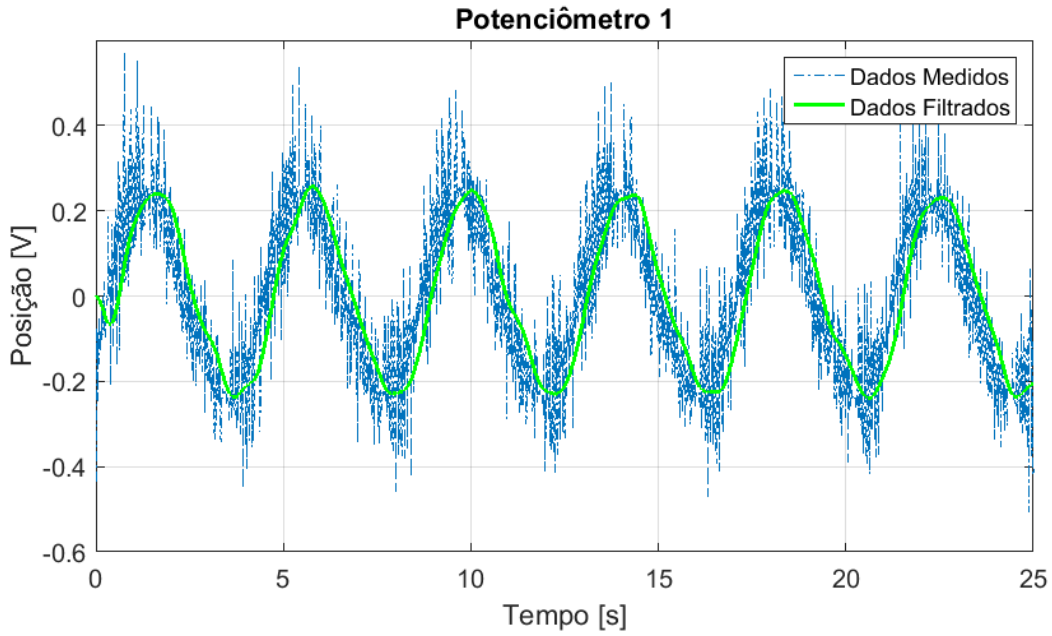


Figura 62 – Comparação entre as leituras do potenciômetro 1 para uma entrada senoidal de frequência 1.5 rad/s, com elas filtradas pelo filtro Elíptico

Conclui-se que o filtro Elíptico projetado, tem um atraso menor que 0.15 s, marginalmente menor que o filtro Butterworth e similar ao Chebyshev Tipo II. Os resultados em termos de filtragem e distorção do sinal são muito parecidos aos do filtro Chebyshev Tipo II. Este resultado é similar para os outros potenciômetros.

5.4 Estratégias de Controle Projetadas

Sabendo-se a estrutura de controle, o modelo da planta e o filtro de medida se pode, finalmente, considerar o projeto do controlador. Tomando em conta as informações da seção 3.4, se considerou projetar quatro tipos de controladores. Dois usando as técnicas clássicas, controlador PID com sintonização; e projeto de controlador usando lugar das raízes. Os outros dois são usando técnicas modernas de controle por realimentação de estados.

A estrutura de controle a ser usada é no espaço das juntas com controladores independentes para cada junta, como se explicou na seção 5.1. Os modelos usados para fazer o projeto se encontram na equação 5.2, para o controle moderno se transforma eles para espaço de estados como explicado na seção 5.2.2

Entre os filtros projetados se optou por usar o filtro de 1ª Ordem para o projeto inicial, pois era mais fácil de ser considerado no projeto de controladores por lugar das raízes, e para os métodos de controle moderno.

Todos os controladores foram projetados usando as ferramentas do MATLAB e do Simulink. Para testar seus desempenho se usou dois tipos de testes. Um teste foi com uma

referência do tipo degrau a ser seguida, este teste se usou para verificar que a malha de controle conseguia acompanhar a referência em um tempo determinado. Este foi realizado *offline* numa simulação usando o Simulink e tendo como planta o modelo identificado.

O outro teste foi uma senoide, este se usou para comparar os atrasos entre os diferentes controladores para seguimento de uma trajetória, e também para garantir que a ação de controle não ultrapasse os limites físicos dos atuadores. No entanto, devido ao rescalamento do sinal de controle quando se faz a implementação no ControlDesk, este teste só é valido quando usado em tempo real, ou seja, com a plataforma se movimentando.

Este ultimo teste, portanto, se fez no ControlDesk com a plataforma funcionando. Não serão mostrados os resultados deste teste neste capítulo, mas serão discutidos no capítulo 7, depois de se explicar a implementação prática dos controladores no capítulo 6.

O projeto dos controladores se fez considerando seguimento de referências do tipo degrau, com os seguintes requisitos:

- Garantir seguimento de referências do tipo degrau, com erro zero em regime permanente;
- Não pode existir *overshoot* maior que 5%, isto pode ser traduzido como um coeficiente de amortecimento de no mínimo 0.7;
- Tempo de resposta de no máximo 2 s, idealmente menor que 1 s;
- Todos as malhas de controle devem ter o mesmo tempo de resposta.
- Se deve usar o menor ganho possível, não mais do que 3, para evitar saturar os atuadores.

Cabe lembrar que não se pode usar como referência a ação de controle usada para um degrau, pois na prática o simulador de voo nunca realizara movimentos bruscos que possam representar uma referência degrau a ser seguida pelo controlador. Isto é garantido sempre e quando o Modelo Dinâmico da Aeronave e o Algoritmo de Movimento funcionem corretamente.

Também se considerou o efeito dos ruído no sistema de controle e o primeiro teste, novamente, com ruído gerado computacionalmente, para observar a eficácia do filtro, e por consequência, seu efeito na malha de controle. Isto foi feito usando o bloco `Band-Limited White Noise`, onde se consegue simular o efeito de ruído branco de determinada energia no modelo.

5.4.1 Sintonização PID

A sintonização do PID se fez criando a malha de controle no Simulink, e se empregava o bloco `PID`. Neste bloco é possível usar o *toolbox* do MATLAB, chamado de *PID Tuner*. Este *toolbox* permite projetar e sintonizar os parâmetros do PID numa malha de controle. Se usa o próprio modelo do Simulink para simular, iterativamente, o comportamento do malha de controle, para cada mudança feita nas constantes do PID.

A vantagem deste método está na sua simplicidade, uma vez que o controlador já tem uma forma determinada e se tem liberdade de sintonizar apenas 4 parâmetros do controlador, o ganho proporcional, integrativo, derivativo e o filtro derivativo usado. Também não é necessário tomar em conta a função de transferência do filtro, no sentido que não é necessário mudar a estrutura do controlador, mas basta apenas sintonizar novamente os valores com um filtro diferente.

A desvantagem deste método é que se torna uma tarefa tortuosa, sintonizar os parâmetros do PID para os 6 atuadores tenham o mesmo tempo de resposta.

No projeto inicial do PID se usou como requisito um tempo de resposta de 1.4 s para todas as juntas. Como a planta já possui um integrador, não era necessário usar o termo integrativo do PID, então se determinou como zero o ganho deste termo, tornando efetivamente os controladores PID em PD. O bloco `PID` do Simulink, implementa um controlador PID com a seguinte configuração:

$$\boldsymbol{P} + \boldsymbol{I}\frac{1}{s} + \boldsymbol{D}\frac{\boldsymbol{N}}{1 + \boldsymbol{N}\frac{1}{s}}. \tag{5.11}$$

Os parâmetros determinados para cada um dos parâmetros do PID se mostram na tabela 3, estes valores foram usados na projeto inicial, considerando o uso do filtro de 1ª Ordem.

Atuador	P	I	D	N.F
1	1.695	-	0.056	1.369
2	2.352	-	0.038	1.428
3	1.777	-	0.036	1.444
4	1.770	-	0.005	1.265
5	1.586	-	0.126	1.539
6	1.739	-	-	-

Tabela 3 – Ganhos dos controladores PID projetados inicialmente

Como foi dito anteriormente, se fez um teste com degrau unitário como referência para garantir que os requisitos são atendidos, o resultado deste teste se pode ver na figura 63. No teste feito se observou que os tempo de resposta não era idêntico em todas as juntas, mas era muito parecido, e se optou por não mudar a sintonização. A não ser que no teste prático esta diferença cause uma defasagem visível entre os atuadores.

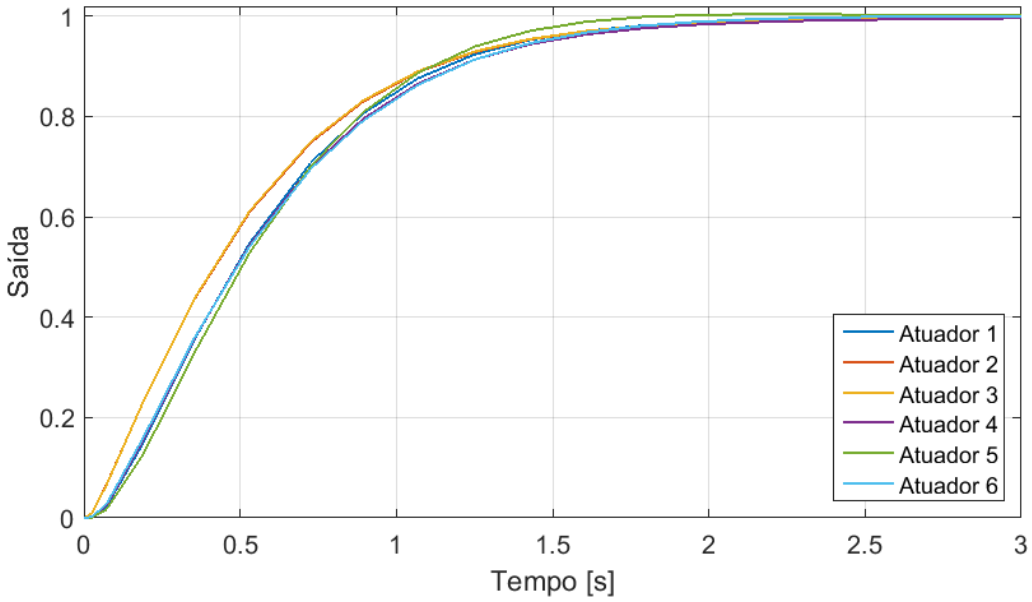


Figura 63 – Comparação das respostas dos controladores projetados por sintonização PID de todos os atuadores para uma referência do tipo degrau unitário, sem ruído de medida presente

Se fez um segundo teste com a mesma entrada, desta vez se adicionou ruído na medida, e se observou sua influência no sistema. Os resultados deste teste se observam na figura 64. Se observa que a adição do ruído não afeto muito o desempenho dos controladores. Lembrando que os dois testes foram feitos com o filtro de medida presente.

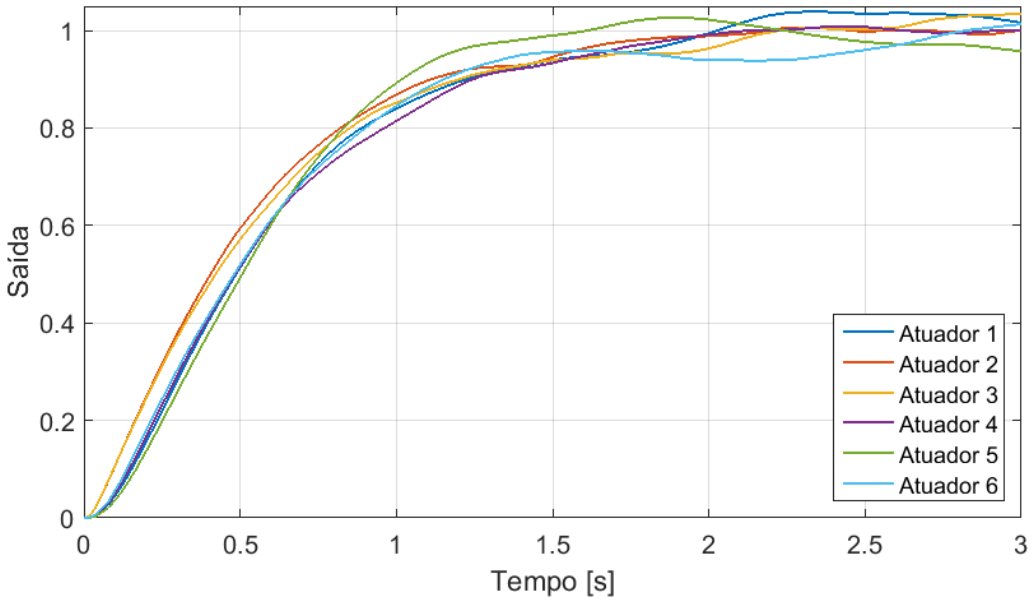


Figura 64 – Comparação das respostas dos controladores projetados por sintonização PID de todos os atuadores para uma referência do tipo degrau unitário, com ruído de medida presente

É importante ressaltar que, embora nos testes práticos se mostrem a saída filtrada, ou seja, a leitura dos potenciômetros depois de passar pelo filtro de medida, nestes testes simulados é interessante olhar a saída depois da planta e antes da adição do ruído artificial, pois isto dá uma certa expectativa do comportamento real das juntas ativas, isto é, se é possível que exista *overshoot* ou uma diferença significativa na velocidade de resposta.

5.4.2 Lugar das Raízes

Para fazer o projeto do controlador usando a metodologia de lugar das raízes, se usou o *toolbox* do MATLAB chamado *Control System Designer*, também conhecido como *SISO Tool*, ele pode ser usado com o comando `controlSystemDesigner` ou `sisotool`.

Este *toolbox* permite projetar e sintonizar os parâmetros de uma malha de controle. A característica mais importante é a possibilidade de agregar ou tirar polos e zeros do controlador, e ver como estes afetam os polos de malha fechada do sistema, assim como ajustar o ganho do controlador para movimentar os polos de malha fechada. O *toolbox* também é capaz de gerar gráficos de reposta degrau da malha, e observar em tempo real como as mudanças dos parâmetros da malha afetam à resposta. [26]

A desvantagem deste método é a dificuldade de sintonizar os parâmetros do controlador para as 6 juntas, pois se deve garantir que todas as juntas tenham o mesmo tempo de resposta. Isto decorre a que a sintonização de faz visualmente ao invés de matematicamente.

Outra desvantagem deste método é que a ordem do filtro dificulta consideravelmente a sintonização do controlador, pois o filtro agrega mais polos que devem ser alocados para locais que garantam os requisitos do projeto.

Os controladores projetados para cada um dos atuadores, considerando um filtro 1ª Ordem (equação 5.5), são os seguintes:

$$\begin{aligned}
C_1 &= \frac{31.9(s+300)}{(s+25)(s+150)}, \\
C_2 &= \frac{27.79.9(s+300)}{(s+20)(s+150)}, \\
C_3 &= \frac{70.25(s+300)}{(s+50)(s+150)}, \\
C_4 &= \frac{34.9(s+300)}{(s+25)(s+150)}, \\
C_5 &= \frac{47.5(s+300)}{(s+40)(s+150)}, \\
C_6 &= \frac{201(s+60)}{(s+30)(s+150)}.
\end{aligned} \tag{5.12}$$

Se pode observar que os controladores não possuem polos integradores, isto se deve a que a planta já é integradora e portanto não é necessário isto para garantir seguimento de referências do tipo degrau com erro zero em regime permanente. Estes controladores garantem que a tempo de assentamento (tempo de 5%) seja de 0.85 s para todos as malhas, assim como um overshoot sempre menor a 5%.

O lugar das raízes, com os zeros (circunferências vermelhas) e polos (simbolo "x" em vermelho) do sistema inicial, e os polos de malha fechada (pontos em magenta) para o Atuador 1, com seu controlador respectivo, se observa na figura 65. Não será apresentado o mesmo diagrama para o resto dos atuadores, pois todos eles são muito similares ao já apresentado.

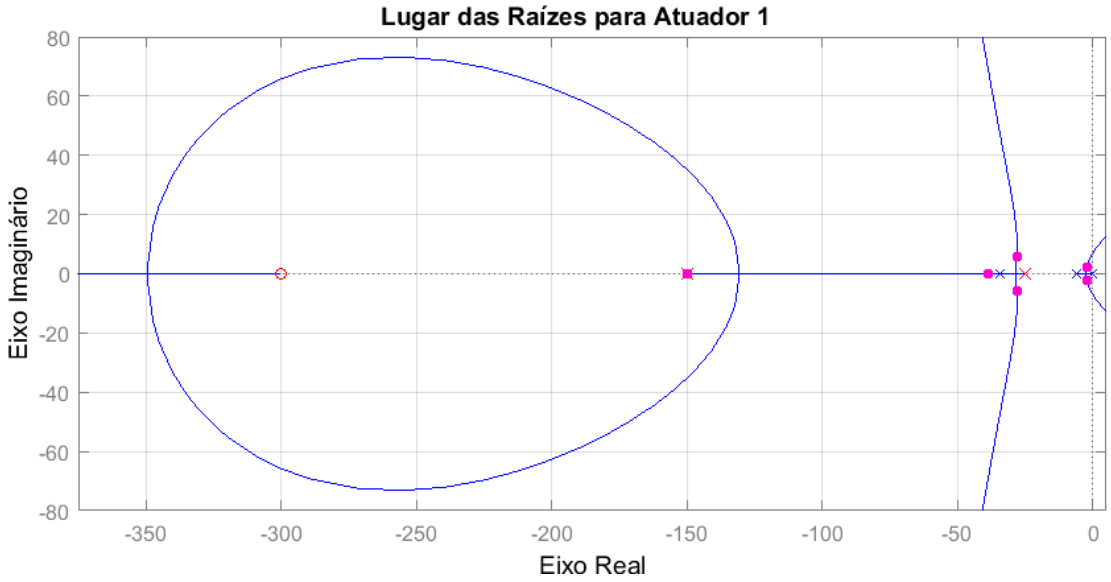


Figura 65 – Lugar das raízes e polos de malha fechada para o Atuador 1

O resultado destes controladores para uma entrada degrau unitária se pode ver na figura 66. Se pode apreciar como os controladores conseguem ter um seguimento muito parecido para as diferentes plantas identificadas de cada junta.

Se fez um segundo teste com ruído para verificar, o comportamento da malha, os resultados se podem observar na figura 67. Obteve-se uma boa resposta, e se observou que apenas existe menos ruído de baixa frequência no movimento das juntas prismáticas, que o PID.

Neste resultado se observou que o atuador 6 sobrepassou o limite de overshoot de 5%, causado pelo ruído de medida presente. Se optou por não mudar o ganho do controlador do atuador 5, pois se precisaria ajustar os ganhos dos outros controladores para manter o mesmo tempo de resposta.

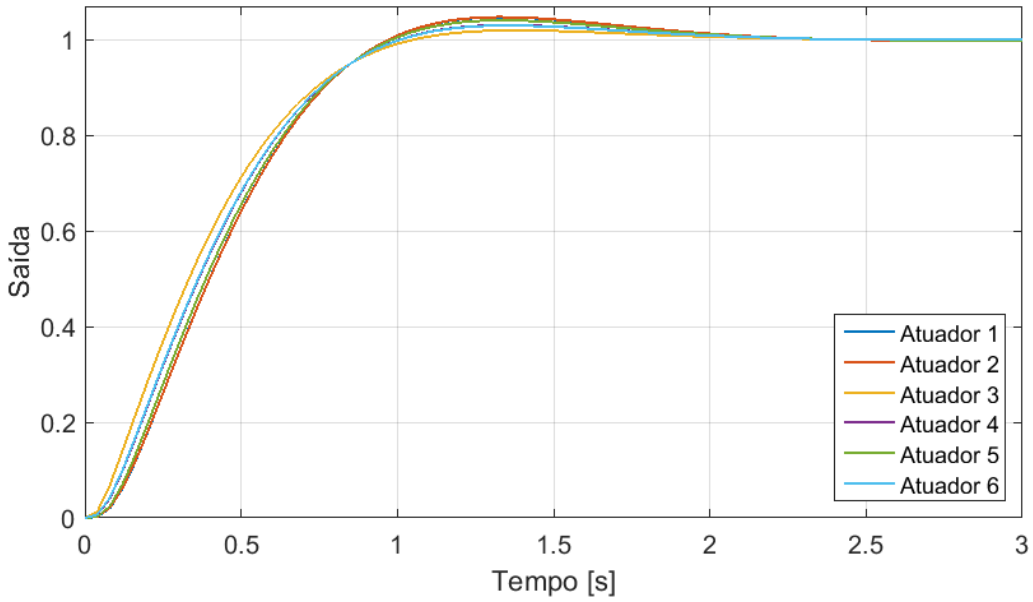


Figura 66 – Comparação das respostas dos controladores projetados pelo lugar das raízes de todos os atuadores para uma referência do tipo degrau unitário, sem ruído de medida presente

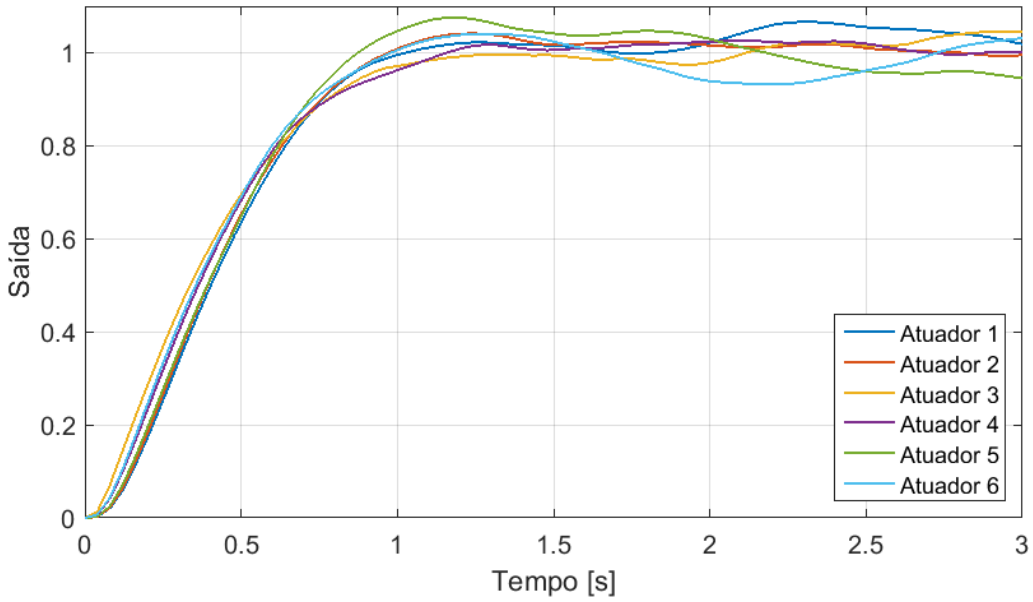


Figura 67 – Comparação das respostas dos controladores projetados pelo lugar das raízes de todos os atuadores para uma referência do tipo degrau unitário, com ruído de medida presente

5.4.3 Alocação de Polos

Esta metodologia de projeto de controle foi apresentada no seção 3.4.2.2. Como se disse nesta seção, para poder realizar o controle do sistema usando realimentação de

estados é necessário determinar o valor da matriz de ganho $\mathbf{K}$ do controlador. Esta matriz determina a dinâmica com a qual o erro de seguimento de referência converge para zero.

A dinâmica do erro depende dos autovalores da matriz $\mathbf{A} - \mathbf{B}\,\mathbf{K}$, ou seja, é possível determinar a velocidade com a qual o erro converge para zero através do valor escolhido para a matriz de ganho. Lembrando que, o seguimento de referência do tipo degrau é garantido usando a estrutura apresentada na figura 25, pois a planta a ser controlada já é integradora.

A metodologia usada para determinar o valor desta matriz que é denominada de alocação de polos, pois a matriz $\mathbf{K}$ é tal que se garante quer os polos (autovalores) de $\mathbf{A} - \mathbf{B}\,\mathbf{K}$ sejam os escolhidos. É possível realizar isto no MATLAB usando o comando `place(v,sys)`, onde `v` é o vetor contendo os valores desejado para os polos de malha fechada, condizente com a ordem do sistema (quantidade de estados) `sys`.

A metodologia de alocação de polos, garante que é possível alocar os polos para qualquer região desejada sempre e quando o sistema seja controlável, ou seja, que a matriz de controlabilidade (equação 3.15) tenha posto completo.

Se pode fazer esta verificação usando os comandos `ctrb(A,B)` e `rank(X)` do MATLAB, o primeiro calcula a matriz de controlabilidade para o par de matrizes `A` e `B`, e o segundo calcula o posto de uma matriz `X`.

Se verificou que todos os modelos tinham matrizes de controlabilidade de posto completo, e portanto todos os sistemas eram controláveis. Através de testes preliminares, se escolheu que os polos de malha fechado como os seguintes valores $[-5.61 \pm 2.9\,i, -9, -8]$. Os polos complexos foram usados tentando igualar aos polos dominantes de malha fechada, que se tinham ao se projetar o controlador por lugar das raízes (figura 65).

5.4.3.1 Observador de Estados

Antes de se implementar o controle por realimentação de estados é necessário ter uma medida destes e como se explicou na seção 4.4, não se tem medidas físicas dos estados, portanto é indispensável usar um observador de estados para fazer uma estimação deles.

Similarmente ao caso do lugar das raízes, para projetar os controladores no espaço de estados se deve tomar em conta que os polos agregados pelo filtro de medida usado. Sendo assim o projeto dos observadores também deve tomar em conta o filtro de medida usado, isto implica que quanto maior a ordem do filtro maior será a ordem do observador. Portanto o uso dos filtro de mais alta ordem apresentados neste capítulo, dificultam o projeto dos observadores.

Para realizar o projeto dos observadores se usou as teorias apresentadas na seção 4.4. Se pode modificar a dinâmica do observador, isto é, a velocidade com a qual o erro de estimação tende a zero, modificando os autovalores da matriz $\mathbf{A}^T - \mathbf{C}^T\mathbf{K}_e$. Isto se logra

modificando a matriz de ganho dos observadores $\mathbf{K}_e$. Para fazer isto se usou a mesma estrategia usado para projetar o controlador, o método de alocação de polos.

Lembra-se que os valores das matrizes **A** e **C** são iguais aos modelos usados para fazer o projeto dos controladores, modelos estes que foram determinados pelas técnicas explicadas na seção 5.2.2.

Assim se usou o mesmo comando do MATLAB, `place`, neste caso se usa as matrizes **A** e **C** transpostas. Similar ao caso da matriz de ganho do controle, se pode alocar os polos do observador em qualquer local, se e somente se, a matriz de observabilidade (equação 4.33) tem posto completo.

Se consegue calcular a matriz de controlabilidade pelo MATLAB usando o comando `obsv(A,C)`. Sendo assim se verificou, para cada um dos modelos das juntas, se tinham a propriedade de observabilidade. Isto foi constatado para todos os modelos.

Inicialmente se usou 4 polos "quase iguais", pois por problemas numéricos o comando `place` do MATLAB não consegue calcular uma matriz capaz de alocar todos os polos para um mesmo valor. Os polos desejados para o observador foram todos aproximadamente iguais a -30, este valor é pelo menos 3 vezes maior que o polo mais rápido do sistema de controle. Isto garante que o observador terá uma influencia mínima na saída do sistema.

Com o observador e controlador projetado se realizaram simulações no Simulink para testar seu desempenho. Na figura 68 se mostram o resultado deste teste quando não se tinha ruído de medida.

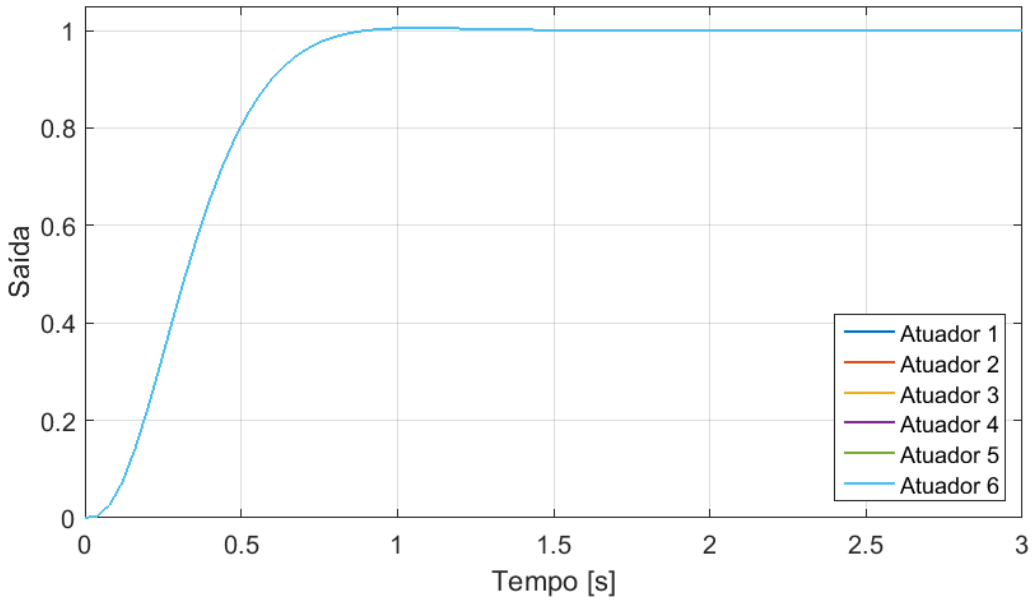


Figura 68 – Comparação das respostas dos controladores projetados pela alocação de polos de todos os atuadores para uma referência do tipo degrau unitário, sem ruído de medida presente

Neste teste se pode verificar como, o comando `place` consegue alocar os polos de malha fechada no mesmos lugares, garantindo que todos as juntas tenham o mesmo comportamento. Tanto é que não se pode observar uma diferença entre as respostas dos 6 atuadores.

Para estes controladores se teve um tempo de resposta de aproximadamente 0.68 s, com uma overshoot menor a 1%. Se exalta que com o uso desta técnica o projeto dos controladores se torna muito mais fácil, pois o projeto inteiro se faz usando apenas um comando do MATLAB para cada um dos atuadores. Isto simplifica bastante a tarefa de projetar e testar vários controladores com polos de malha fechada diferentes.

Além disto, sempre e quando se usem os mesmos polos, se garante facilmente que todas as juntas tenham o mesmo tempo de resposta. No entanto, é mais difícil visualizar qual que será a ganho de total do controlador, pois o controle é realizado por uma matriz de ganhos, comparado às metologias de controle clássicas.

Se fez um segundo teste para avaliar a presença do ruído de medida, isto se vê na figura 69.

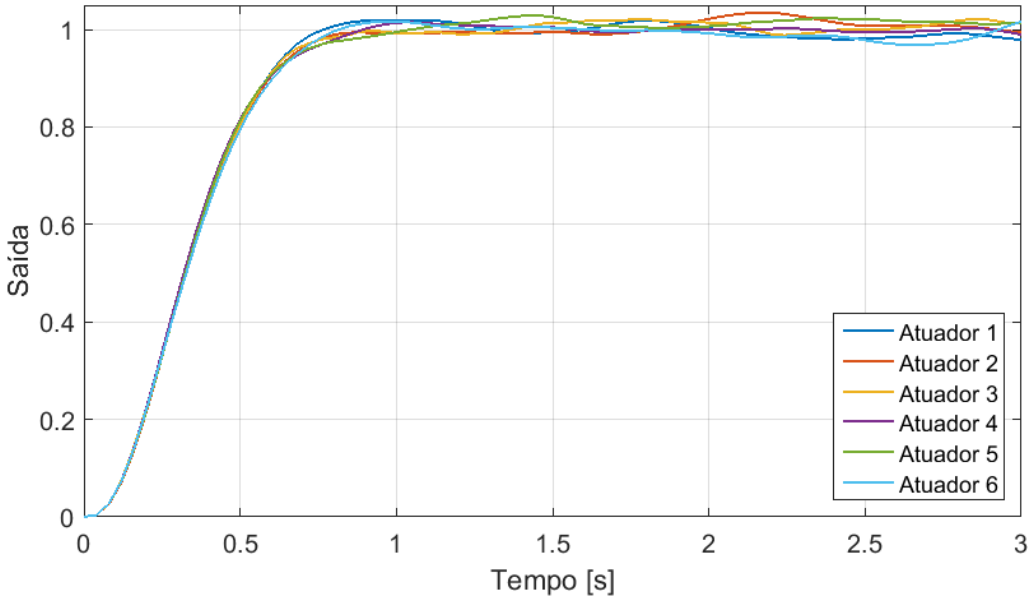


Figura 69 – Comparação das respostas dos controladores projetados pela alocação de polos de todos os atuadores para uma referência do tipo degrau unitário, com ruído de medida presente

Neste teste se verificou que a presença do ruído quase não afeto o desempenho do controlador, tanto é que, no regime transitório todas as juntas se comportavam de maneira idêntica e apenas no regime permanente se podia ver o efeito do ruído. Também se verificou que o *overshoot* não ultrapassou o limite estabelecido pelos requisitos.

Como nota final, se esclarece que se tentou não usar o filtro de medida, e usar o observador como forma de filtrar o ruído de medida. Isto tornaria o projeto de controle

muito mais simples, caso se usa-se filtros de ordem alta. No entanto os observadores não eram capazes de eliminar o ruído satisfatoriamente, ao invés que o filtro de 1ª Ordem, e se abandono esta ideia.

5.4.4 Projeto do Regulador Linear Quadrático

Como se viu na seção 3.4.2.3, este tipo de controlador também usa a realimentação de estados, mas muda o método pelo qual é calculada a matriz de ganho. Este método tenta calcular uma matriz tal que a função de custo 3.19 seja minimizada.

O MATLAB consegue calcular tal matriz usando o comando `lqr(sys,Q,R)`, onde `sys`, é modelo no espaço de estados do sistema; e `Q` e `R` são as matrizes respectivas da função de custo.

Inicialmente se determinou como matrizes identidade para estas matrizes e se foi iterativamente ajustando seus valores, com o uso de simulações no Simulink. Os valores que se determinaram foram os seguintes:

$$\mathbf{Q} = \begin{bmatrix} 10500 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0.001 \end{bmatrix}, \quad \mathbf{R} = 100. \tag{5.13}$$

Se observar que se tem valores cada vez menores conforme se avança nos estados, isto é devido a que a saída é o primeiro estados, e os outros são as derivadas dele, assim se quer estabilizar mais rapidamente o primeiro estado, que consequentemente significa que os outros estados já se estabilizaram. Se usou uma valor maior do que 1 na matriz $\mathbf{R}$, para evitar que o ganho do controlador aumente sem limites.

Um dos atuadores teve que usar um matriz $\mathbf{Q}$ um pouco diferente, pois tinha um desempenho ruim para a outra. Isto provavelmente é por causa que este atuador está mais desgastado e portanto seu modelo difere do resto das outras juntas. Esta matriz é apresentada a seguir.

$$\mathbf{Q}_2 = \begin{bmatrix} 10500 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 0.0005 \end{bmatrix}, \tag{5.14}$$

Assim como no caso da metologia de alocação de polos, se teve uma maior dificuldade para tentar realizar o projeto, quando se tinha um filtro de medida de ordem alta.

O resultado destes controladores para uma entrada degrau unitária se pode ver na figura 70.

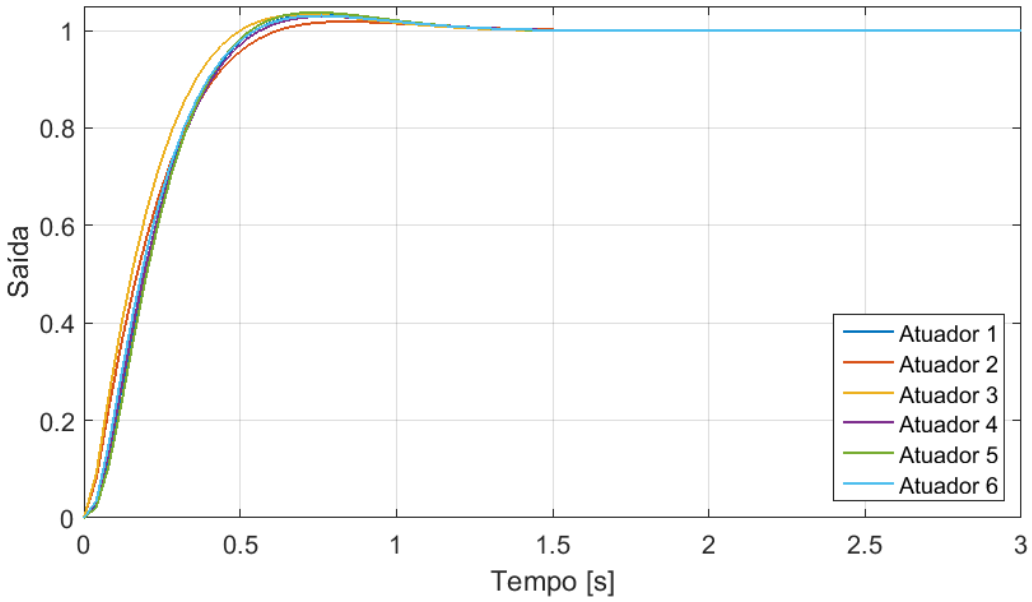


Figura 70 – Comparação das respostas dos controladores LQR de todos os atuadores para uma referência do tipo degrau unitário, sem ruído de medida presente

Neste resultado se observa que quase todos os atuadores tiverem uma resposta igual, excetuando o atuador 3. Outra característica que se observa é o comportamento da saída, com um tempo de resposta de aproximadamente 0.45 s e *overshoot* menor a 5%.

Da mesma forma que os anteriores se fez um segundo teste, adicionando ruído fictício, os resultados se podem ver na figura 71. Neste resultado se constatou que não existia *overshoot* maior do que 5% em nenhum dos atuadores.

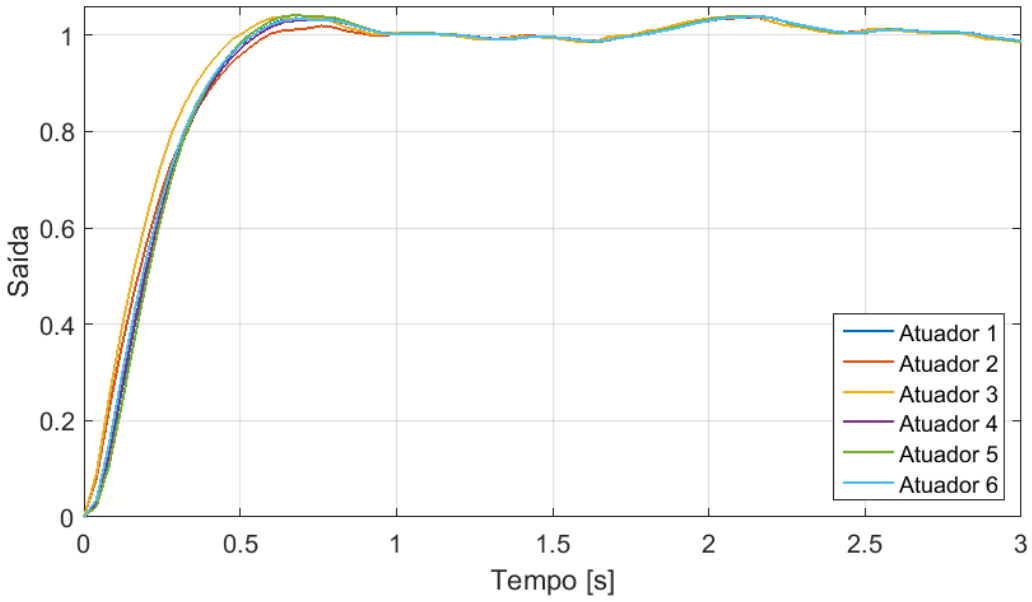


Figura 71 – Comparação das respostas dos controladores LQR de todos os atuadores para uma referência do tipo degrau unitário, com ruído de medida presente

5.5 Resumo do Capítulo

Neste capítulo se explicou a estrutura de controle que foi escolhida, **controle no espaço das juntas descentralizado**, e o porque desta escolha. A característica mais importante é que as referências usadas para seguimento são no espaço das juntas com o uso da cinemática inversa, e que o controle se da por separado em cada junta, considerando vários sistemas SISO, onde o acoplamento entre as juntas é considerado como uma perturbação.

Para realizar o projeto dos controladores se precisa de um modelo no qual basear-se. O modelo usado foi identificado a partir de uma **resposta ao degrau** do **sistema integrativo** de cada junta, que tinha como **saída uma rampa**. Se usou as técnica discutida no capítulo anterior e se validou a identificação comparando com o valor real filtrado e derivado.

Se verificou que existe ruído de medida que impactava negativamente o controle do robô, e para isso se projeto quatro tipos de filtro passa baixa, um de 1ª Ordem, um Butterworth de 4ª Ordem, um Chebyshev Tipo II de 4ª Ordem e um Elíptico de 3ª Ordem.

Entre os filtros projetados, **se optou por usar o de 1ª Ordem**, pois este tinha um bom desempenho para a baixa ordem que apresentava. Isto acaba simplificando o projeto da maioria dos controladores.

Se realizaram quatro tipos de projeto de controlador, em cada um se usava um método diferente para projetar o controlador, se uso sintonização PID, lugar das raízes, alocação de polo e LQR. Entre estes quatro se discutiu as vantagens e desvantagens de cada metodologia.

Entre os quatro, o projeto mais **simples** era do PID, onde se tinha apenas 3 graus de liberdade para definir o controlador e também seu projeto **independia do filtro**, pois apenas se ajustava os parâmetros dele para levar em conta os diferentes filtros. No entanto era difícil conseguir que todas as juntas tenham o **mesmo tempo de reposta**, além de ser **trabalhoso** sintonizar os 6 controladores necessários.

O lugar das raízes tinha a vantagem de ser bastante **flexível** e se tinha uma boa estimativa do comportamento do controlador no momento de realizar o projeto. Fazer com que todas as juntas tivessem o mesmo tempo de resposta não era difícil mas era trabalhoso. Porem este é o controlador mais **complicado** de projetar quando se tinha filtro de ordem maior, e ainda era **trabalhoso** realizar o projeto para as 6 juntas.

No controle moderno se usou a alocação de polos, este método era **muito simples** de ser projetado, com a possibilidade de se trocar os polos de todos os polos facilmente. No entanto, realizar o projeto quando se tem filtros mais alta ordem é também mais complicado.

Outra desvantagem é a necessidade de usar um observador de estados. O próprio projeto do observador também fica mais complexo quanto maior a ordem filtro usado, e ele sempre tem que ser de 2 a 5 vezes mais rápido que a malha de controle.

O ultimo controlador projetado é o LQR, neste se observou que era complicado ponderar corretamente as matrizes da função de custo, o que tornava este método mais difícil de usar que outro método para o controle por realimentação de estados. No entanto se teve um desempenho melhor que de alocação de polos.

6 Implementação dos Controladores

Usando as teorias apresentadas nos capítulos 3 e 4, e levando em conta as analises discutidas no capítulo 5, se pode começar a explicar o processo de implementação os controladores que resolveram o problema definido na seção 3.1.

A ordem apresentada dos tópicos não reflete à ordem cronológica na qual foi feita a implementação. A metodologia seguida para implementar o sistema de controle é a seguinte:

1. Definir a estrutura da malha de controle;
2. Obter um modelo adequado com o qual se realizará o projeto do controlador;
3. Definir e projetar o filtro da medida, caso precisar;
4. Baseado no modelo e considerando a presença do filtro, ou a falta de, projetar o controlador, se precisar projetar elementos adicionais, por exemplo, observador;
5. Simular e analisar o desempenho do controlador na simulação. Caso seja suficiente, prosseguir, se não, corrigir ou descartar;
6. Implementar no Simulink, e usar o modelo no ControlDesk;
7. Realizar testes em tempo real e analisar o desempenho; opcional gravar os dados da simulação para análises mais profundas usando outros *softwares*.

O sistema de controle completo implementado no Simulink é apresentado na figura 72. Ele é dividido em vários subsistemas: geração de trajetórias $(x, y, z, roll, pitch, yaw)$ da aeronave; cinemática inversa; ajuste de referências; controladores; envio de sinais; leitura dos sinais; e filtros de medida.

Como se demostro no capítulo 5, neste trabalho se usou o MATLAB e Simulink para realizar o projeto e simulação de todos os elementos do sistema (cinemática inversa, filtro de medida, controladores, observadores de estados). Isto facilito a implementação no ControlDesk, pois apenas era necessário integrar eles em um único modelo para ser compilado e usado no ControlDesk.

Ainda, como se usou a mesma estrutura de controle (no espaço das juntas), e o uso de subsistemas, se facilitou a troca dos controladores e filtro pois apenas era necessário trocar os blocos contidos nos subsistema determinado, deixando o resto do modelo igual.

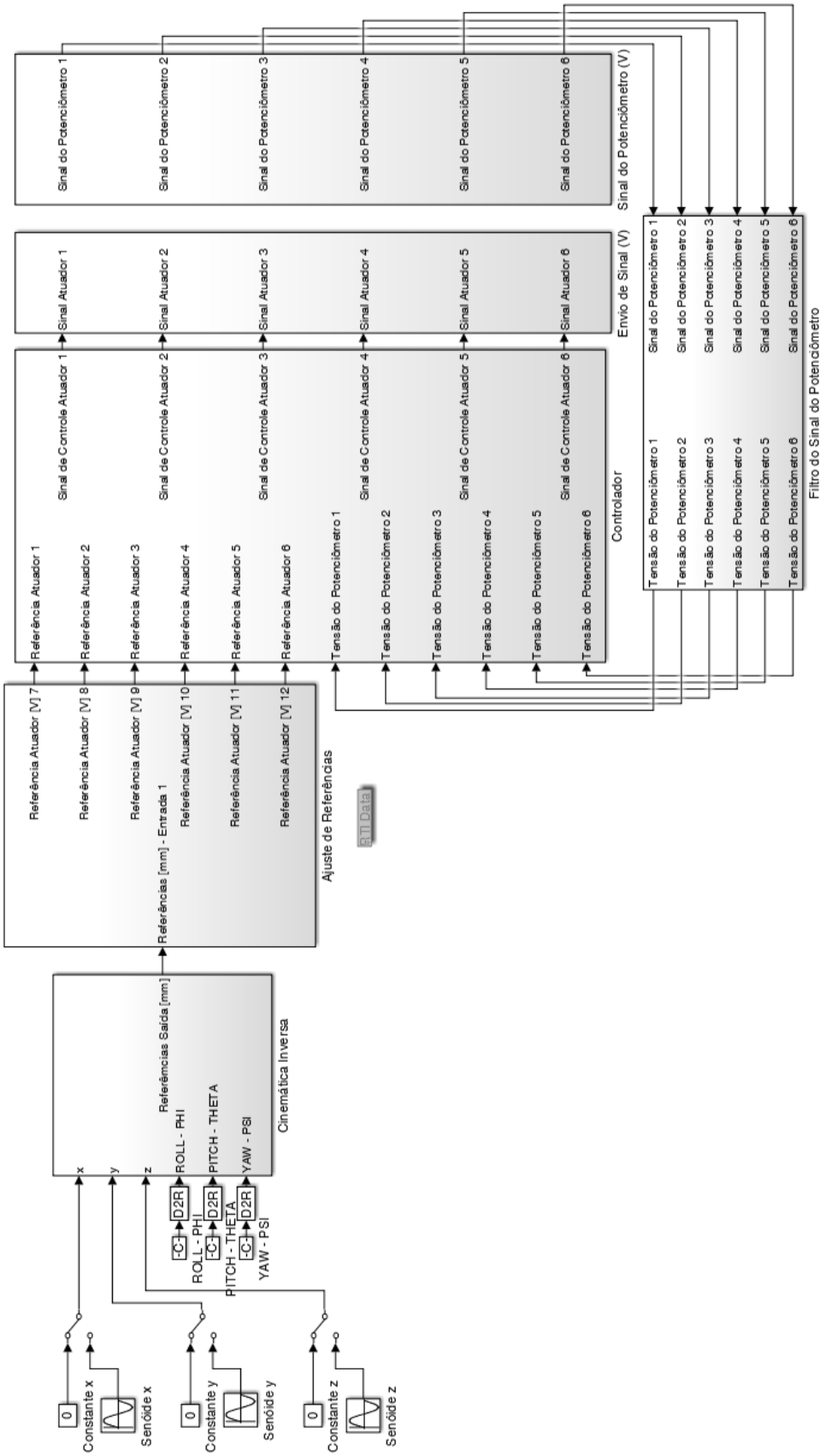


Figura 72 – Estrutura de controle no espaço das juntas implementado no Simulink

Como as próprias simulações para testar o projeto do filtros e controladores se fez usando o Simulink. Isto facilitou a implementação necessária com as bibliotecas da dSPACE, para realizar os testes práticos.

Inicialmente, na seção 6.1, se dará um resumo rápido para lembrar quais são os elementos da malha de controle e onde estes se encontram no mundo real. Nesta mesma seção se explicará resumidamente as ferramentas usadas para poder usar os elementos do painel I/O no Simulink, e o que se deve fazer para poder usar o modelo do Simulink no ControlDesk

A implementação da estrutura de controle no Simulink será discutida na seção 6.2, a estrutura da figura 72, é igual para todos os controladores projetados, apenas é necessário mudar os elementos internos do subsistema, o mesmo se aplica para o filtro.

Com isso a implementação dos controladores se explicará em uma seção separada (6.3). Na seção subsequente 6.4, será apresentado o supervisório criado no ControlDesk e algumas das ferramentas usadas implementá-lo, além das usadas para poder gravar dados que possam ser analisados posteriormente.

6.1 Elementos da Malha de Controle

Como se mencionou anteriormente, se optou por usar a **estrutura de controle descentralizada no espaço das juntas**, como mostrado na figura 73. Nela, se pode identificar conceitualmente os elementos da malha e quem representa eles na implementação prática, como previamente dito na seção 2.1.

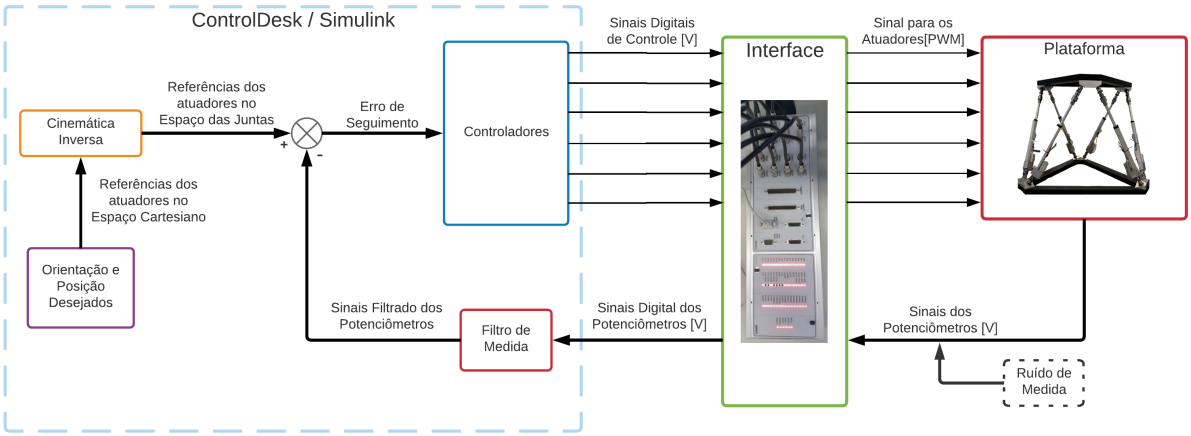


Figura 73 – Diagrama da malha de controle e seus elementos

No momento de realizar os testes práticos o Algoritmo de Movimento ainda não estava implementado, e portanto apenas se pode usar nos teste trajetórias artificiais como senoides. Não é interessante o uso de degraus, pois no funcionamento típico de um

simulador de voo, e de uma aeronave real, não existem trajetórias que contenham uma descontinuidade, como a existente nos degraus.

6.1.1 Ferramentas usadas para a Implementação prática

Praticamente todas as simulações feitas para testar o desempenho dos controladores foram feitas no **Simulink**. Algoritmos com maior complexidade, como o **Algoritmo de Movimento** e o calculo da **Cinemática Inversa** foram implementados na linguagem de *script* do **MATLAB**. Estes algoritmos foram, então, transportados para a ambiente de simulação do Simulink, denominado de *modelo*, usando o bloco *MATLAB Function*, que permite o algoritmo execute a cada passo de simulação. Lembrando que o Algoritmo de Movimento não foi implementado pelo aluno, e estava sendo desenvolvido paralelamente.

Como mencionado já na seção 2.1, modelos criados no Simulink podem ser "compilados" em programas de linguagem `C` ou `C++` , para que sejam usados por outros *softwares*. Isto se faz usando o comando *Build Model*, presente no botão mostrado na figura 74.

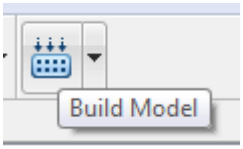


Figura 74 – Botão para "compilar" o modelo do Simulink (*Build Model*)

Quando se compila os modelos, o programa gera vários arquivos de diferentes extensões, no mesmo diretório do modelo. Um destes arquivos tem a extensão `.sdf`, e pode ser usado no ControlDesk da dSPACE. Isto significa que o ControlDesk é capaz de simular o mesmo modelo criado no Simulink, mas com a vantagem de poder criar telas supervisórias, onde se pode mostrar variáveis e gráficos mais facilmente.

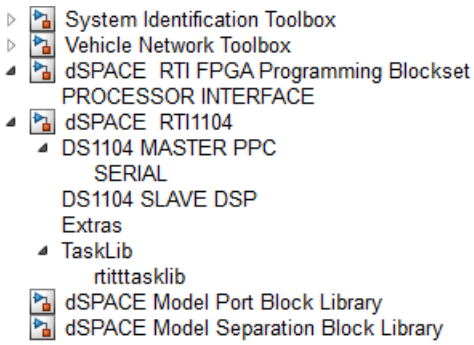


Figura 75 – Bibliotecas da dSPACE para uso da DS1104 e do painel I/O

Ainda, se for usado as bibliotecas do Simulink para usar a placa controladora DS1104 e o painel I/O conectado a ela, como se vê na figura 75, se pode implementar modelos para usar em tempo real, com as saídas e entradas disponíveis do painel I/O.

Uma boa analogia para descrever o uso do Simulink com o ControlDesk, é o *software* da *National Instruments*, *LabView*. Se tem a facilidade de projetar controladores fácil e intuitivamente, usando o Simulink e MATLAB; e usar o ControlDesk para implementar e supervisar em tempo real estes.

Uma clarificação importante é que embora os controladores foram projetados em tempo continuo, a sua implementação é tecnicamente em tempo discreto. Apesar disso, na prática os controles tem pouca diferença pois o tempo de amostragem usado tanto no Simulink como no ControlDesk, é de 0.01 s, que é muito superior às dinâmicas esperadas dos atuadores, assim se pode considerar que o sistema funciona em tempo continuo sem causar erros grosseiros.

É importante que os tempos de amostragem do modelo usado nos dois programas sejam iguais, e, no caso do Simulink, que este seja fixo (por padrão é usado tempo um variável).

6.2 Implementação da Estrutura de Controle no Simulink

O modelo criado, no Simulink,para implementar a estrutura de controle no espaço das juntas se pode observar na figura 72. O sistema se dividiu em subsistemas para simplificar a aparência e facilitar a edição deste modelo. Nota-se que existe um bloco chamado `RTI Data`, ele é necessário para pode usar a DS1104, e sempre é adicionado aos modelos do Simulink por padrão, quando se instala as bibliotecas da placa.

6.2.1 Referência e Cinemática Inversa

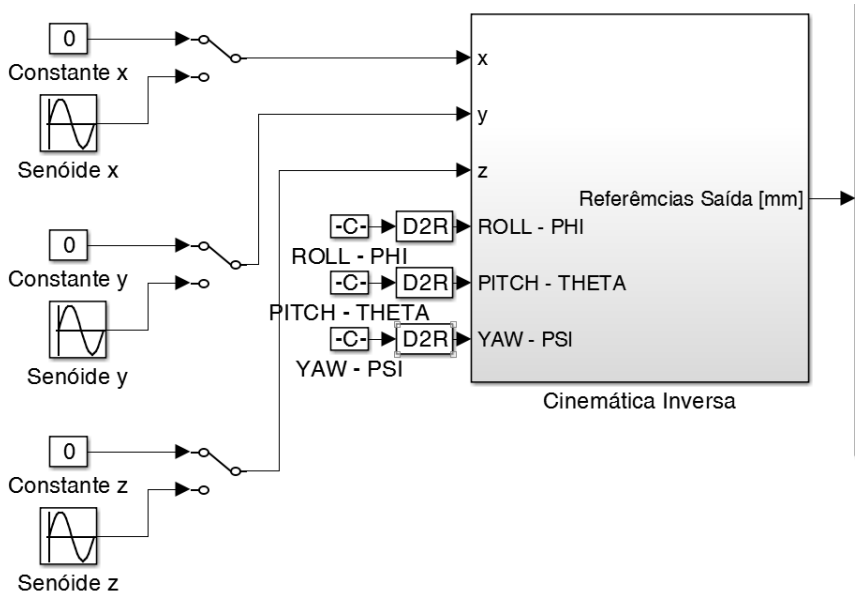


Figura 76 – Detalhe das Referências e Cinemática Inversa do modelo implementado

Estes subsistemas definem os vetores de referência que precisam ser seguidas pelo Mecanismo de Movimento. Eles podem ser vistos na figura 76.

Inicialmente se define o vetor de referência $[\mathbf{t}, \mathbf{\Theta}] = [x, y, z, \phi, \theta, \omega]$. O modelo tem uma chave de seleção onde se pode trocar entre usar uma entrada de valor constante ou senoide para a referência de posição da plataforma $\mathbf{t} = [x, y, z]$. Posteriormente esta chave pode ser manipulada pelo ControlDesk, e assim se pode mudar em tempo real a referência que se quer usar para o experimento. O mesmo pode ser feito para a orientação $\mathbf{\Theta} = [\phi, \theta, \omega]$, mas por enquanto não é necessário.

As referências de posição são dadas em mm em relação à posição inicial, podendo ser positivas a negativas. As de orientação são dada em graus em relação a orientação inicial, mas são convertida pelo bloco `D2R`, para radianos para que possam ser usados pelo algoritmo que calcula a cinemática inversa. Lembrando que se usa o padrão da aviação para posição e orientação, visto na figura 17.

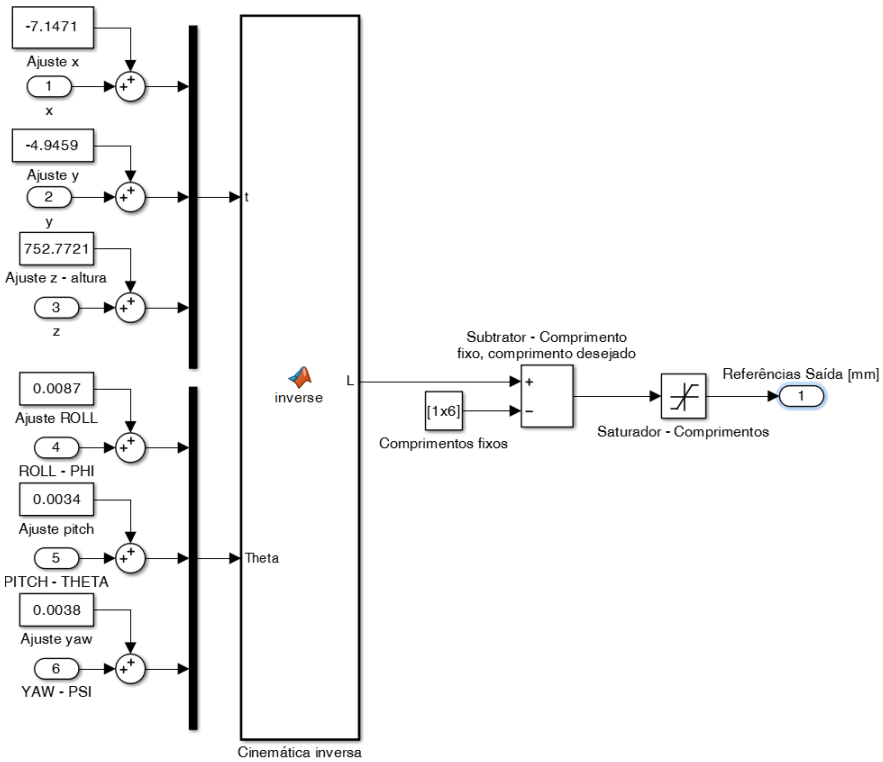


Figura 77 – Subsistema `Cinemática Inversa`

Dentro do subsistema `Cinemática Inversa`, se encontra a solução da cinemática inversa, ele contem os blocos da figura 77.

Inicialmente se faz um ajuste das referências para tomar em conta a posição e orientação inicial que se quer no mecanismo. E depois se faz o mesmo para as referências no espaço das juntas. Finalmente se usa um saturador para evitar que se passe uma referência maior da necessária. Assim o subsistema `Cinemática Inversa`, retorna os valores de referência para os atuadores em mm.

A cinemática inversa foi implementada usando a linguagem de *script* do MATLAB e usada no Simulink com o emprego do bloco do tipo `MATLAB Function` (`inverse`). Este *script* do MATLAB será apresentado a seguir.

```
function L = inverse(t,Theta)

BB =[-507.4000 -499.9000  181.8000  307.2000  311.3000  195.1000;
      -69.6000   72.6000  471.3000  406.1000 -397.8000 -457.8000;
       63.6000   60.4000   61.9000   64.6000   63.5000   61.1000];

PP =[-258.3 -266.4 -133.5  413.4  412.3 -142.8;
     -316.1  317    392.7   70.6  -65.6 -396.6;
      -68    -64.2  -64.6  -67.7  -60.8  -66.7];

phi    = Theta(1);
theta  = Theta(2);
psi    = Theta(3);

R = [cos(psi)*cos(theta),-cos(phi)*sin(psi)+cos(psi)*sin(theta)*sin(phi),cos(psi)*cos(phi
    )*sin(theta)+sin(psi)*sin(phi);
     cos(theta)*sin(psi),cos(psi)*cos(phi)+sin(psi)*sin(theta)*sin(phi),cos(phi)*sin(psi)
         *sin(theta)-cos(psi)*sin(phi);
     -sin(theta),cos(theta)*sin(phi),cos(theta)*cos(phi)];

q = R*PP;
S = q + [t t t t t t]  - BB;

L = [norm(S(:,1)) norm(S(:,2)) norm(S(:,3)) norm(S(:,4)) norm(S(:,5)) norm(S(:,6))]  ;
end
```

O algoritmo 1 visto na seção 5.1.2, foi usado como base para fazer este *script*, no entanto se substitui o uso do *for*, pelo uso de matrizes para calcular os 6 vetores $\mathbf{S}_i$ e concatená-los em uma matriz única `S`. Igualmente os vetores constantes $\mathbf{p}_i$ e $\mathbf{b}_i$, são representados na matriz `PP` e `BB`.

6.2.2 Ajuste de Referência

Este subsistema se encarrega em ajustar as referências dadas em mm pela cinemática inversa em valores condizentes de tensão em V que serão enviadas ao *drivers* dos motores através do painel I/O. O subsistema `Ajuste de Referências` contem os blocos mostrados na figura 78

O ajuste se dá por meio de uma relação linear, ou seja, aplicando um ganho e um *offset*. Assim a referência passa, de ser mm, a ser V, com o intuito de poder ser comparada com as leituras feitas no potenciômetro, também em V.

Esta parte da implementação não foi realizado pelo aluno, mas por um aluno de mestrado que também trabalha na plataforma. Isto também é valido para os ajustes feitos antes de realizar a cinemática inversa, e também aos feitos ao enviar e receber sinais do painel I/O.

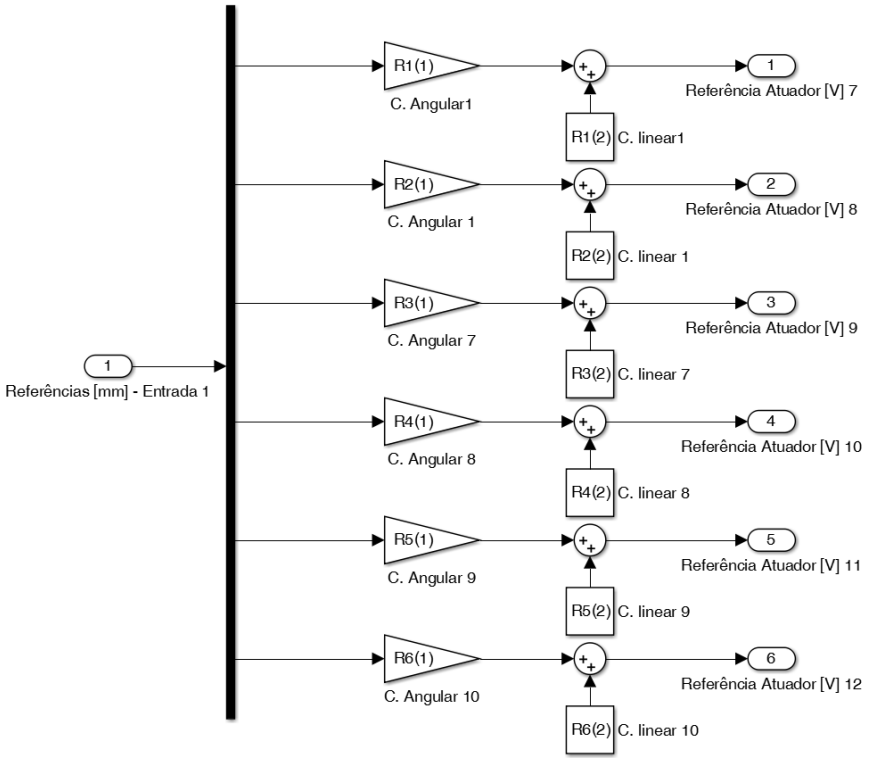


Figura 78 – Subsistema `Ajuste de Referências`

6.2.3 Controlador e Envio do Sinal

Figura 79 – Detalhe dos subsistemas `Controlador` e `Envio de Sinal (V)`

O subsistema `Controlador` recebe as referências ajustadas em V, assim mesmo recebe o sinal filtrado da realimentação. Com estes dois sinais se é possível calcular o

sinal de controle em tensão, que será enviada ao subsistema `Envio de Sinal (V)`, que se encarrega de enviar este sinal ao painel I/O. Estes podem ser vistos na figura 79.

O bloco `Controlador` é um pouco diferente para cada uma dos projetos dos controladores e, por isso, serão mostrados individualmente mais a adiante neste capítulo. Se nota que o subsistema `Envio de Sinal (V)` não possui saída, contudo na prática a sua saída se da através do Painel I/O, que enviará os sinais de controle calculados aos *drivers* dos motores para que sejam convertido em um sinal PWM. Portanto se pode considerar que o sinal de controle calculado é uma "referência" para os *drivers* ajustarem o *duty cycle* do sinal PWM aplicado no estator dos motores das juntas prismáticas.

Para poder enviar os sinais pelo painel é necessário usar um dos blocos das bibliotecas adicionadas pela DS1104, este bloco se chama `DS1104DAC_-Cx`. Este bloco envia o sinal digital da simulação para a DAC, que o transforma em um sinal analógico que pode ser usado pelo *driver*. Antes de enviar é fundamental reescalar o sinal por causa da DS1104, isto se faz usando apenas um ganho (de 0.1).

Isto se pode observar na figura 80. Nesta figura os blocos com DAC escritos são os blocos mencionados, que dão acesso aos conversores, neles se determina qual conversor se quer usar.

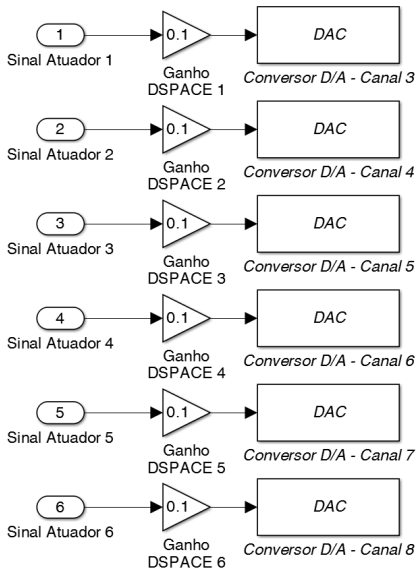


Figura 80 – Subsistema `Envio de Sinal (V)`

6.2.4 Sinal do Transdutor e Filtro de Medida

Para fechar a malha se realimenta o sinal que indica o comprimento da junta prismática. Este sinal se obtém com o uso dos potenciômetros integrados às juntas. Os sinais deles são recebidos pelos conversores ADC do painel I/O, que os convertem a sinais digitais.

Para usar estes sinais no Simulink se usa o bloco da DS1104 `DS1104ADC_-Cx`. Contudo, os primeiros quatro canais usam um mesmo conversor e só são accessíveis pelo bloco `DS1104MUX_ADC`. Nestes blocos se escolhe quais canais se quer usar, e no ultimo no ultimo é necessário separar o sinal usando um *demux*.

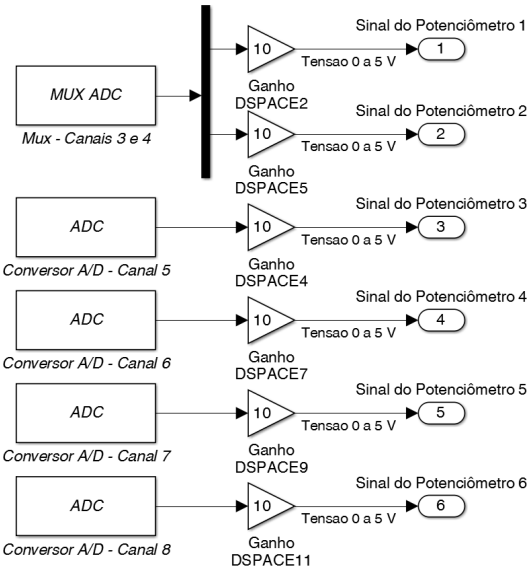


Figura 81 – Subsistema `Sinal do Potenciômetro (V)`

Na figura 81 se pode ver a implementação disto dentro do subsistema `Sinal do Potenciômetro (V)`. Também se reescala o sinal usando ganhos estáticos.

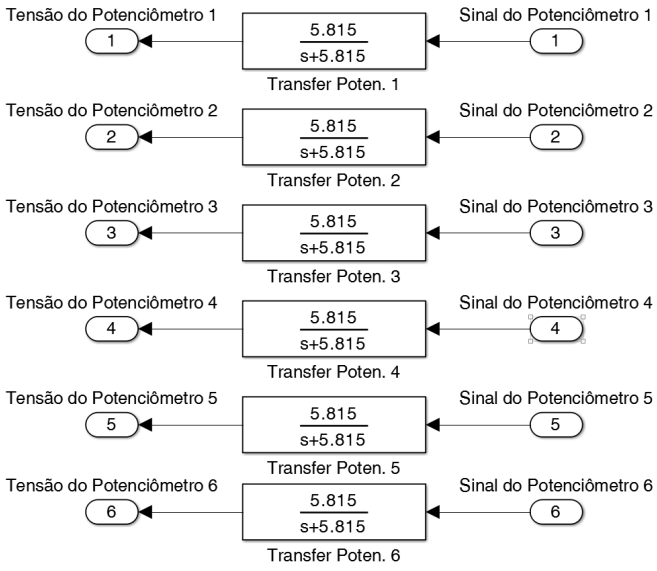


Figura 82 – Subsistema `Filtro do Sinal do Potenciômetro`

O bloco `Filtro do Sinal do Potenciômetro`, contem os filtros que serão projetados, o bloco basicamente passa os sinais do subsistema `Sinal do Potenciômetro (V)`, por funções de transferência que representam o filtro projetado. Estes por sua vez são usado para realimentar o sistema e fechar a malha. Como se conclui na seção 5.3, se usou apenas os filtros de 1ª ordem. O interior deste subsistema se observa na figura 82.

6.3 Implementação dos Controladores

Se pode dividir a implementação dos controladores no Simulink em três tipos, controle PID, controle por lugar das raízes e o controle por realimentação de estados.

Embora se tenha projeta usando 4 métodos diferentes, os controladores modernos usam a mesma estrutura mostrada na figura 25, o que muda entre a alocação de polos e o LQR é a forma como é calculada a matriz de ganhos do controlador. Assim não foi necessário criar estruturas diferentes quando se usou as técnicas modernas, mas apenas se trocava a forma do cálculo da matriz. Já os outros dois tipos controladores se implementaram de forma diferente.

6.3.1 Controle PID

Como se explicou na seção 5.4.1, se fez uso do bloco `PID` do Simulink. A implementação para as 6 juntas seguia o mesmo padrão visto na figura 83, apenas mudando os valores das constantes, conforme a tabela 3.

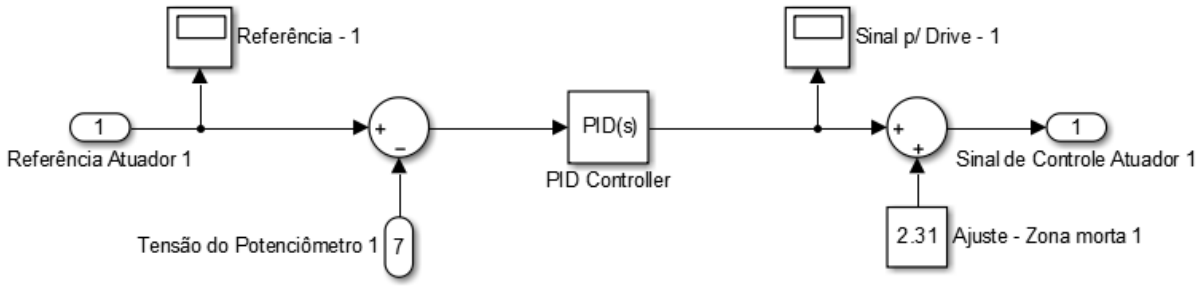


Figura 83 – Implementação do controlador PID no Simulink

Se pode observar que se faz ajuste para tomar em conta a zona morta do motor, adicionando um valor fixo e diferente para cada motor. Isto se deve a que todos os motores possuem uma faixa de tensão inicial na qual eles não conseguem se movimentar, denominada zona morta. A determinação deste valor foi realizada experimentalmente por outro aluno que trabalha no projeto.

6.3.2 Controle por Lugar das Raízes

Para implementar este tipo de controle se usou o bloco `Zero-Pole`, com ele se pode implementar uma dada função de transferência causal, definindo-se os zeros, polos e ganho que ela possui. Como se pode observar na figura 84, se usam vetores (ou matrizes para MIMO) que determinam o valo de cada polo, zero e ganho.

Assim se implementou o controlador determinado pela metodologia de lugar das raízes, com o padrão mostrado na figura 85, aplicado para cada uma das fincões de transferência (equação 5.12) dos controladores, determinada na seção 5.4.2.

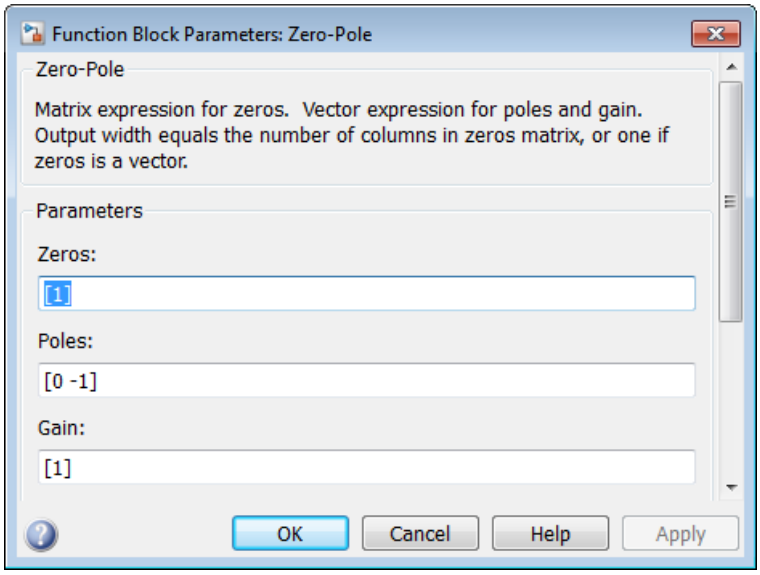


Figura 84 – Informações necessárias para criar um bloco `Zero-Pole` no Simulink

Da mesma forma que na implementação do PID, é necessário ajustar o valor do controle calculado pelo controlador para levar em conta a zona morta de cada atuador, adicionando um *offset*.

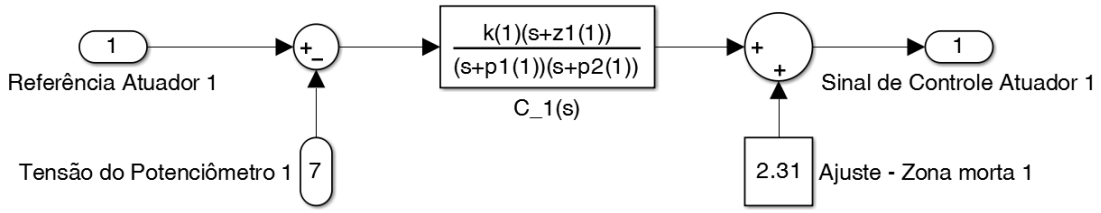


Figura 85 – Implementação do controlador através do Lugar das Raízes no Simulink

6.3.3 Controle por Realimentação de Estados

No caso do controle moderno, apenas muda a forma como a matriz de ganho do controlador é calculada, portanto a implementação no Simulink é igual. Isto se pode constatar na figura 86 e 87.

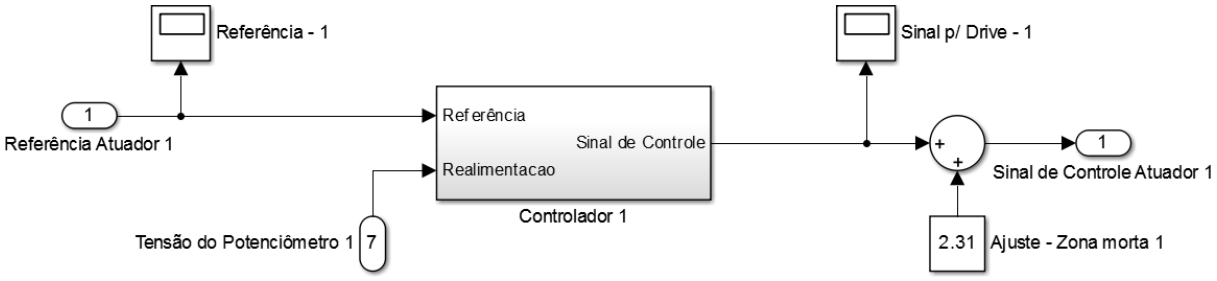


Figura 86 – Implementação dos Controladores por Técnicas Modernas

Igual aos dois controladores já mostrados, se tem o ajuste da zona morta de cada motor.

Como a implementação do controle por realimentação de estados usa mais um elemento, o observador de estados, se optou por englobar todos eles em um subsistema como se pode ver na figura 86. Dentro deste subsistema estão os elementos mostrados na figura 87.

Se pode observar um outro subsistema que representa o observador, e os ganhos do controlador, esta estrutura segue a mesma que estrutura de controle por realimentação de estados para sistemas integrativos, que foi mostrada na figura 25. Neste caso os estados observados (saída do observador) são multiplicados por uma matriz que contem os seguintes ganhos $\begin{bmatrix} 0 & k_2 & \cdots & k_{n-1} & k_n \end{bmatrix}$

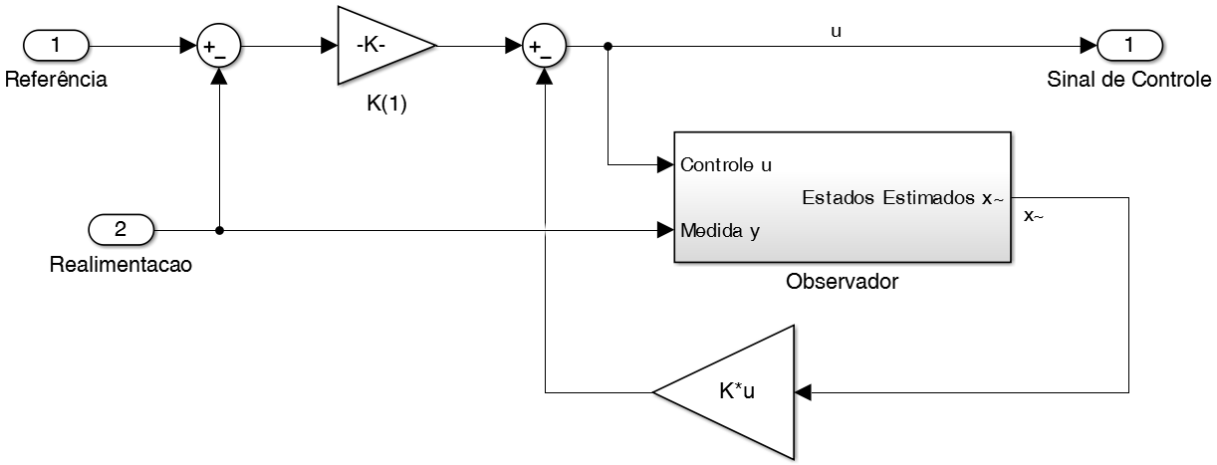


Figura 87 – Implementação dos Controladores por Técnicas Modernas

Esta estrutura segue a mesma que estrutura de controle por realimentação de estados para sistemas integrativos, que foi mostrada na figura 25. Neste caso os estados observados (saída do observador) são multiplicados por uma matriz que contem os seguintes ganhos $\begin{bmatrix} 0 & k_2 & k_3 & k_4 \end{bmatrix}$, enquanto o erro de seguimento é multiplicado por k_1.

Se ressalta que o valor destes ganhos é o mesmo que calculado na matriz de ganhos do controlador, pela técnicas de alocação de polos e LQR, ou seja, $\mathbf{K} = \begin{bmatrix} k_1 & k_2 & k_3 & k_4 \end{bmatrix}$.

Também se destaca que, lembrando que o primeiro estado x_1 é o mesmo que a saída y, se usa o valor real da saída ao invés de usar o valor estimado pelo observador. Isto é devido a que, nos testes iniciais com a plataforma, se obtinha melhores resultados desta forma do que usando a saída estimada pelo observador.

6.3.3.1 Implementação do Observador

Como se discutiu na seção 5.4.3.1 é necessário o uso de um observador pois não se tem medidas dos estados do sistema. O observador implementado se pode ver na figura 88.

Estes são os elementos internos do subsistema do observador da figura 87.

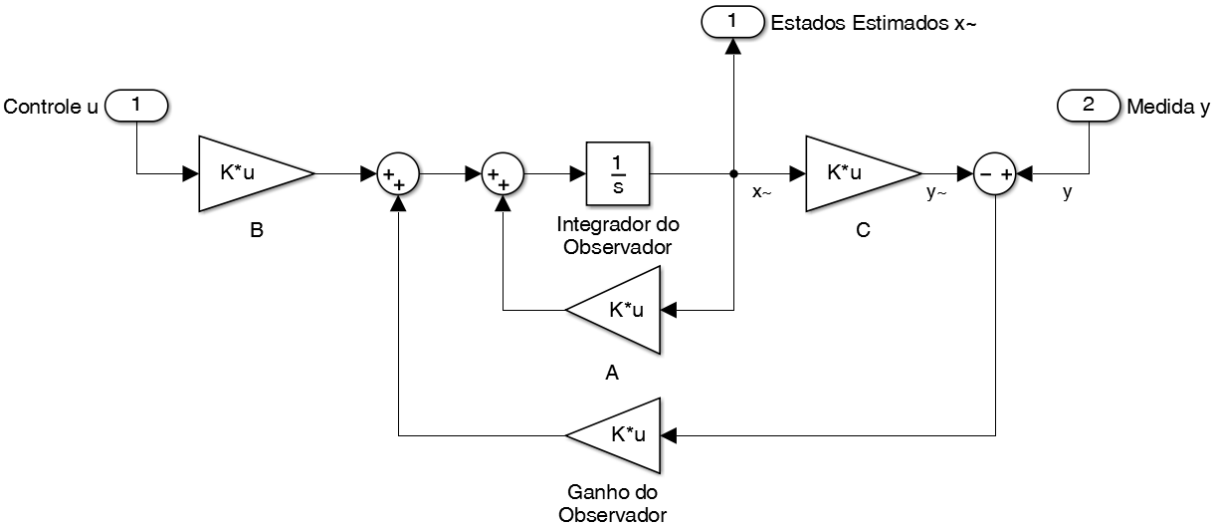


Figura 88 – Implementação dos Observadores de Estado

Dentro do observador se usa os valores das matrizes **A**, **B** e **C**, que representam o modelo identificado da junta respectiva do controlador no qual este observador é usado. Além disto se tem o ganho do observador $\mathbf{K}_e$ multiplicando o erro de estimação. Esta estrutura é a mesma que a mostrada na figura 32.

6.4 Implementação no ControlDesk

Uma vez que se tem o modelo do sistema de controle implementado no Simulink (figura 72) se pode "compilar" ele, usando o comando do Simulink *Build Model* (figura 74). Entre os arquivos gerados por este comando se tem um com a extensão `.sdf`, este é uma arquivo do tipo *Spatial Data File*, que contem as informações do modelo do Simulink. Este arquivo pode ser usado pelo ControlDesk para realizar os testes práticos em tempo real.

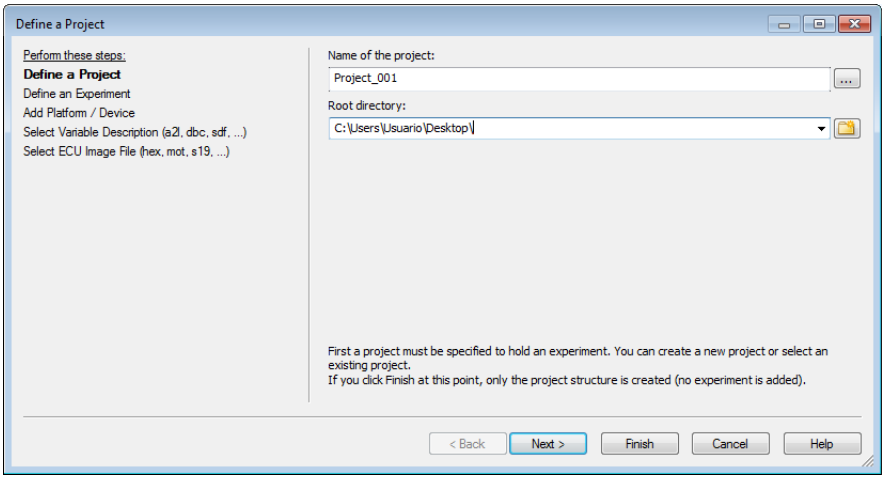


Figura 89 – Primeira etapa para criar um projeto no ControlDesk

Para realizar a implementação no ControlDesk se cria um projeto, usando o *wizard* provido por este, que guia o usuário passo a passo. Inicialmente se escolhe o nome e a localização, no diretório do computador, do projeto. Como se vê na figura 89.

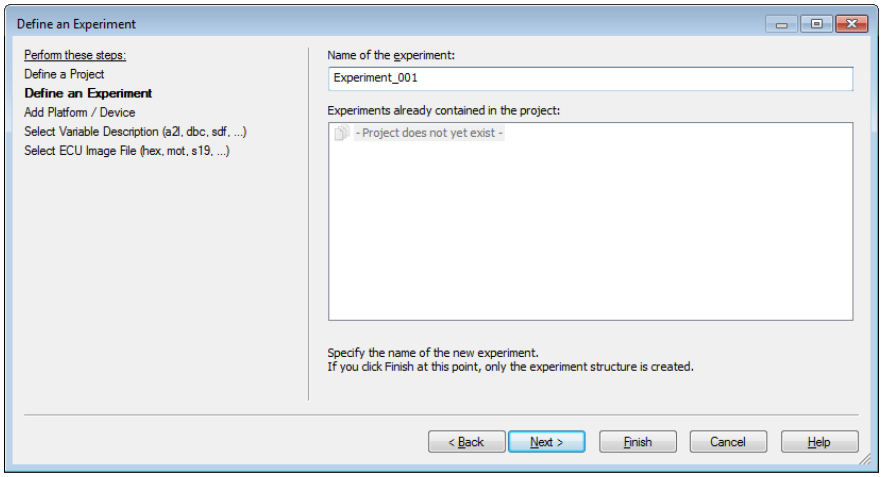


Figura 90 – Segunda etapa para criar um projeto no ControlDesk

A seguinte etapa (figura 90) consiste em definir o nome do "experimento" inicial. Isto se deve a que um mesmo projeto no ControlDesk, pode conter vários "experimentos". Isto é uma forma de poder organizar melhor, caso exista mais de um experimento que pertença a um mesmo projeto.

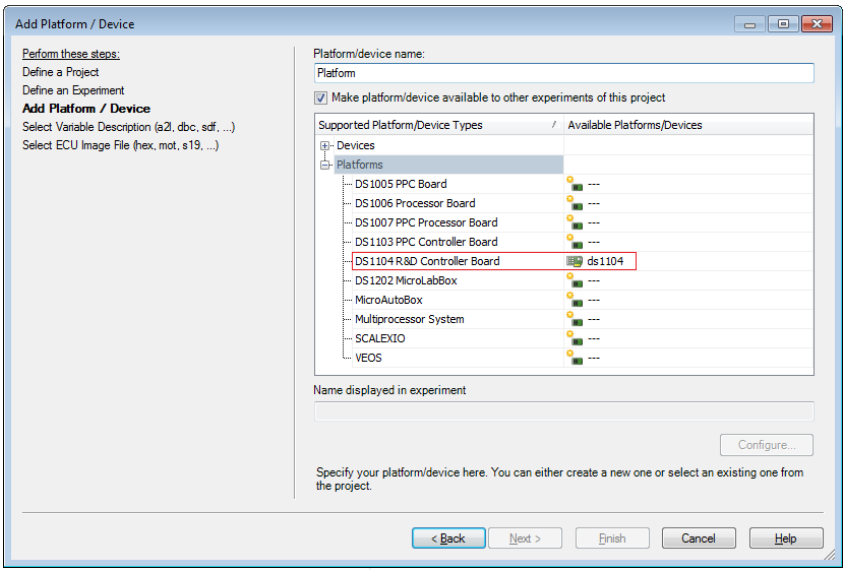


Figura 91 – Terceira etapa para criar um projeto no ControlDesk

O seguinte passo, consiste em escolher e nomear os dispositivos ou plataformas (não confundir com a plataforma robótica) que o ControlDesk deverá usar para o determinado experimento. Isto se pode observar na figura 91, onde se destaca a placa controladora usada, a DS1104.

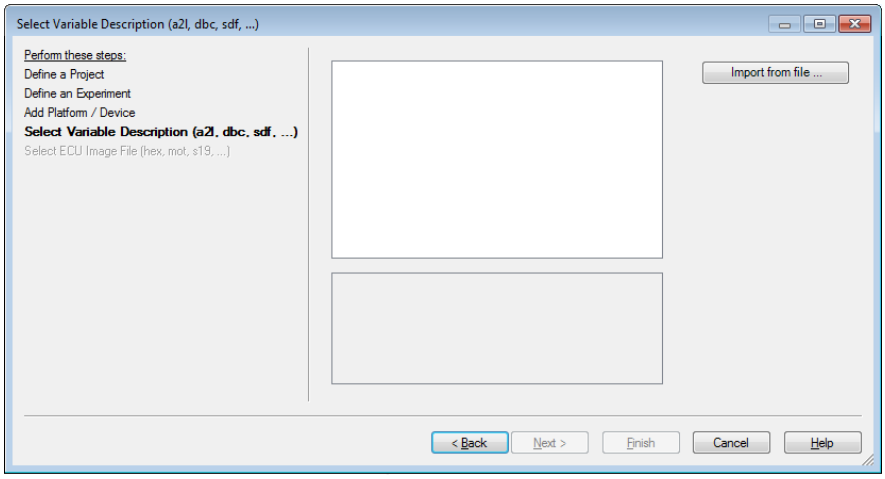


Figura 92 – Quarta etapa para criar um projeto no ControlDesk

Poderia se considerar a quarta e ultima fase, como a mais importante. Nesta fase se importa o arquivo que descreverá o comportamento do ControlDesk, neste caso se usa o modelo compilado do Simulink, o arquivo com extensão `.sdf`, como se observa na figura 92. Com isto se finaliza a criação do projeto no ControlDesk.

Para cada experimento de um projeto no ControlDesk, se cria *layouts*. Estes seriam os supervisores, onde se pode controlar e supervisar o comportamento do sistema, daquele experimento. Ainda é possível criar mais de um *layout* para o mesmo experimento, como por exemplo, no experimento usado para realizar os testes em tempo real (figura 93), se tem 3 *layouts* diferentes, um para supervisar os sinais de saída e controlar os sinais de referência, outro para supervisar os sinais enviados aos *drivers* e mais um para supervisar os estados estimados pelos observadores.

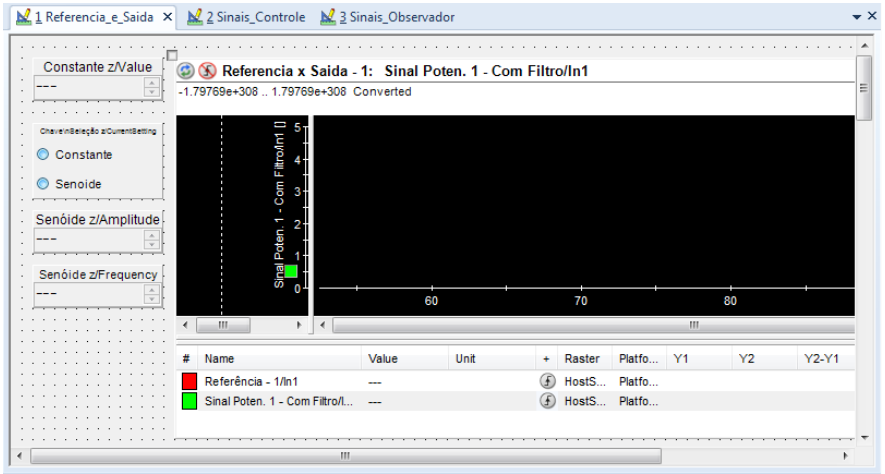


Figura 93 – *Layouts* do experimento usado para realizar a implementação prática no ControlDesk

Dentro dos *layouts* se pode criar vários instrumentos para controlar e supervisar sinais daquele experimentos. Isto se faz usando a janela *Instrument Selector*, mostrada na figura 94.

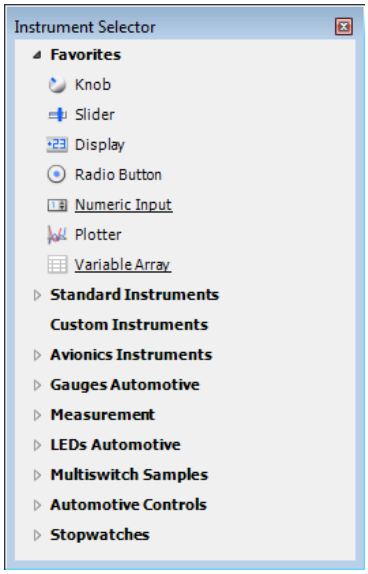


Figura 94 – Janela *Instrument Selector* do ControlDesk, para insertar instrumentos no *layout*

Existem diversos instrumentos passivos e ativos, os passivos são usados apenas para observar variáveis, como por exemplo, o *Plotter*, *Display*, *Variable Array*, etc. Já os ativos são usados para controlar as variáveis do experimento, por exemplo, *Numeric Input*, *Radio Button*, *Slider*, entre outros.

Para insertar os instrumento basta arrastrá-lo para o *layout* desejado. Para determinar qual/is variável/eis este instrumento usará, se usa a janela *Variables* (figura 95). Esta janela contem todas as variáveis existentes neste experimento, tanto as criadas dentro do próprio ControlDesk, quanto às internas do modelo do Simulink compilado.

Ainda é possível acessar os parâmetros internos dos blocos do modelo do Simulink, ou seja, se pode controlar completamente, ou observar, os parâmetros de cada um dos blocos usados no Simulink.

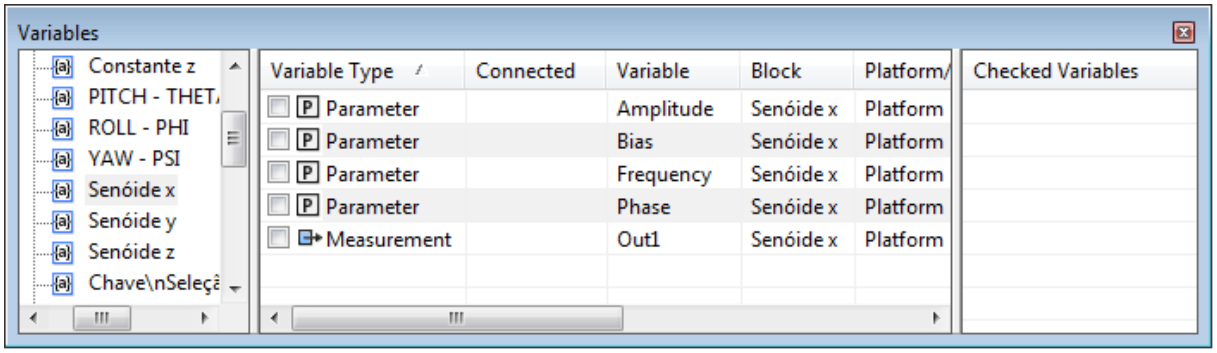


Figura 95 – Janela *Variables* do ControlDesk, para explorar as variáveis do modelo usado para o experimento

Similar à janela de instrumentos, basta arrastrar a variável a qual se deseja vincular ao instrumento. Também é possível fazer o contrário, isto é, arrastrar primeiro a variável

ao *layout* e depois selecionar que tipo de instrumento será vinculado a esta.

No layout da figura 93, por exemplo, se tem o *Radio Button*, vinculado a uma chave seletora, para selecionar o tipo de referência que se quer usar para a coordenada **z** da plataforma móvel (referência no espaço cartesiano), entre referência constante ou senoidal; também se tem 3 *Numeric Inputs*, um vinculado ao valor da constante de referência no eixo **z**, e dois vinculados à amplitude e a frequência (em rad/s) da referência senoidal; por ultimo existe um *Plotter* vinculado ao sinal **filtrado** do potenciômetro, junto com a referência para sua respectiva junta, para traçar um diagrama em tempo real destes.

6.4.1 Gravação e exportação de medidas usando o ControlDesk

Para poder gravar, e poder usar posteriormente, sinais internos e recebidos pela DS1104, se precisa configurar o ControlDesk através da janela *Measurement Configuration*, mostrada na figura 96.

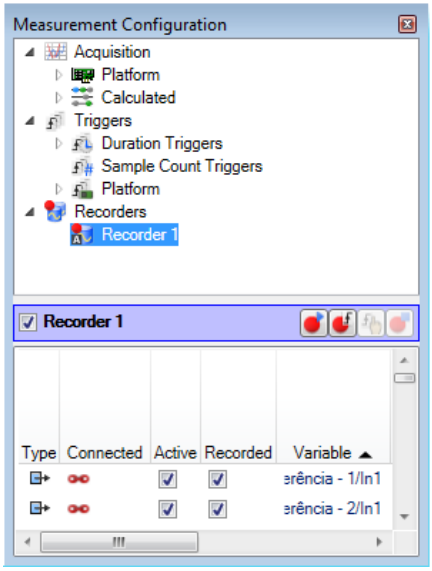


Figura 96 – Janela *Measurement Configuration* do ControlDesk, para configurar a gravação de dados do experimento

Nesta janela se pode configurar quais as variáveis do experimento que se deseja gravar. Estas configurações são feitas em *Recorders*, para acionar estes basta apertar um botão no ControlDesk.

Os *Recorders* especificam quais variáveis deverão ser gravadas, por padrão todas as variáveis vinculadas a um *Plotter* são gravadas. Nele também se determina o nome do arquivo ao qual serão gravados os dados, por padrão a extensão usada pelo ControlDesk é `.idf`, também é possível realizar nomeação automática, onde cada nova gravação usara um padrão e seguira uma ordem numérica.

Adicionalmente se pode optar por realizar uma exportação automática, onde, a cada gravação feita pelo *Recorder*, o ControlDesk exporta os dados para um outro formato,

como por exemplo, `.mat` que pode ser usado pelo MATLAB. Todas estas propriedades do *Recorder* se podem observar na figura 97.

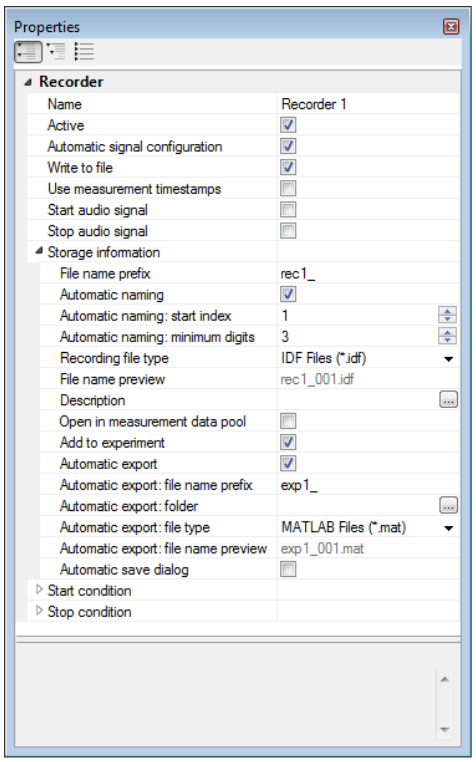


Figura 97 – Propriedades de um *Recorder* do ControlDesk

Se pode configurar *Triggers* para os *Recorders* Os *Triggers* indicam quando que a gravação deverá começar automaticamente, por exemplo, quando aconteça uma mudança de referência, a um tempo específico de simulação, quando ocorra um erro, etc.

As gravações exportadas para o MATLAB tem um formato de *struct* para poder acessar com mais facilidade os vários parâmetros gravados. Um exemplo da *struct* da gravação exportada se mostra na figura 98.

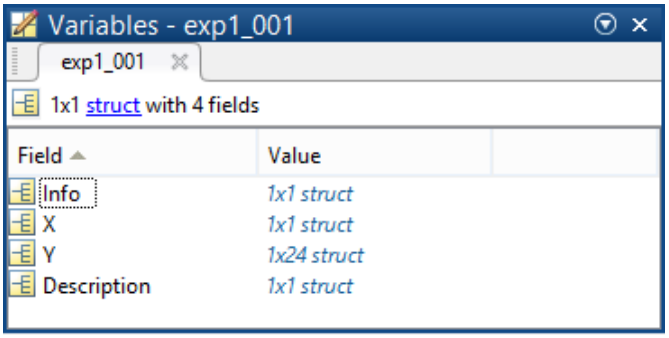
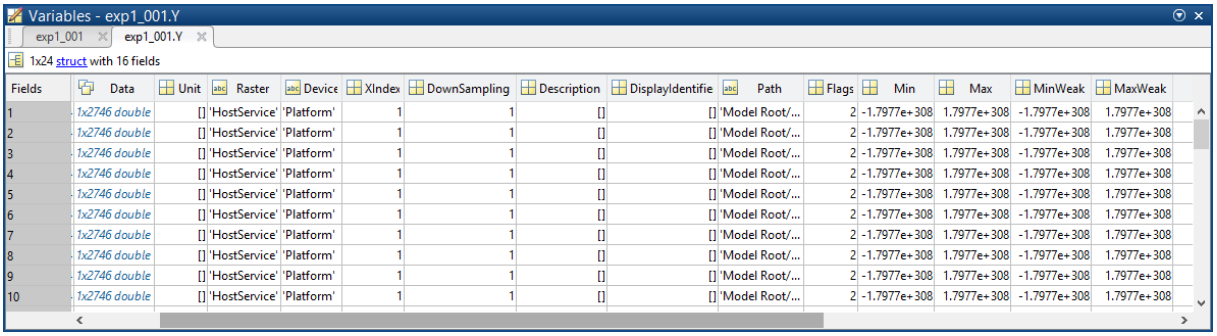


Figura 98 – Dados de gravação do ControlDesk, exportados para uso no MATLAB

A *struct* está conformada por outras *structs* internas. Entre estas, se destaca `X` e `Y`, `X` contem o vetor de tempo da simulação, começando do momento de inicio da gravação até

o final, este vetor tem a mesma frequência de amostragem que a do próprio ControlDesk, que é 0.01 s para este trabalho.

A estrutura `Y` contem os dados e informações adicionais das variáveis gravadas pelo *Recorder*, isto se pode ver na figura 99. O campo `Data` contem os dados, em forma de vetor, com a mesma frequência de amostragem que o ControlDesk. Para poder especificar quais os dados de cada variável se usa o campo `Path` que identifica a origem destes dados, a origem é a mesma que tem na janela *Variables*.



Fields	Data	Unit	Raster	Device	XIndex	DownSampling	Description	DisplayIdentifie	Path	Flags	Min	Max	MinWeak	MaxWeak
1	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
2	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
3	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
4	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
5	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
6	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
7	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
8	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
9	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308
10	1x2746 double	[]	'HostService'	'Platform'	1	1	[]	[]	'Model Root/...	2	-1.7977e+308	1.7977e+308	-1.7977e+308	1.7977e+308

Figura 99 – Dados de gravação do ControlDesk, exportados para uso no MATLAB, detalhe das informações dentro de `Y`

6.5 Resumo do Capítulo

Neste capítulo se apresento a metodologia usada para resolver o problema, ela consta de 7 passos. Resumidamente eles são: Definir a estrutura de controle; Obter um modelo do sistema; Definir o filtro de medida; Projetar o controlador; Simular o controlador; Implementar o controlador no ControlDesk; e Realizar os testes em tempo real.

Os primeiros 5 passos foram explicados no capítulo anterior, a sexta etapa foi explicada neste capítulo. Inicialmente se fez novamente uma explicação geral dos elementos da malha de controle e onde eles se localizam fisicamente, como se vê na figura 73.

O ControlDesk é um *software* usado para implementar sistemas de controle em tempo real usando a placa controladora DS1104. Ele usa como base modelos criados no Simulink para implementar o sistema de controle. O modelo implementado no Simulink se observa na figura 4, ele usa as bibliotecas da dSPACE para fazer uso do painel I/O. Este modelo é compilado usando o comando `Build Model` do Simulink, este gera vários arquivos, entre eles se usa o que tem extensão `.sdf` no ControlDesk

Na seção 6.2 se explica quais os elementos, e como estes foram implementados no Simulink, do sistema de controle. O sistema usa a estrutura de controle no espaço das juntas descentralizado. Ele foi dividido em subsistemas para facilitar a presentação e alteração dele. Dentro deste modelo existe o subsistema `Controlador`, que contem os controladores de cada junta.

Cada um dos controladores projetados é implementado dentro deste bloco, e a implementação de cada um deles é explicada na seção 6.3. No caso do controle por realimentação de estados os observadores também são implementados dentro deste bloco.

Finalmente na seção 6.4 se expõe os passos necessários para poder criar um projeto no ControlDesk, que ira implementar vários experimentos para cada um dos controladores diferentes. Estes experimentos usam os modelos compilados do Simulink, e neles se criam supervisores. Destaca-se que tanto o modelo do Simulink quanto o supervisório no ControlDesk, devem ter o mesmo período de amostragem, que é de 0.01 s para este projeto.

As ferramentas para criar o supervisor também são explicadas nessa seção, assim como as configurações que devem ser feitas para poder gravar dados e exportar os resultados dos experimento em arquivos `.mat` que possam ser usado posteriormente para análise.

Estes resultados são apresentados no capítulo a seguir, para cada um dos controladores projetados. A análise destes foi feita com o uso do arquivo exportado no MATLAB.

7 Experimentos: Resultados e Análises

Neste capítulo serão apresentados os experimentos, para verificar o desempenho do sistema de controle e dos diferentes controladores que se projetaram e explicaram no capítulo 5. Para realizar este testes foi usado a implementação descrita no capítulo 6. Se usou o Simulink para criar o Sistema de Controle, este modelo foi usado pelo ControlDesk para implementar o sistema supervisório do Sistema de Movimento parcial.

O Algoritmo de Movimento não foi usado para a realização destes experimentos, pois ele ainda não tinha sido aperfeiçoado. Isto quer dizer que se usavam referências genéricas para testar o funcionamento do Sistema de Controle.

Lembra-se que o Sistema de Controle possui uma estrutura de controle no espaço das juntas descentralizada. Também o filtro usado nestes experimentos foi o filtro de 1ª Ordem, devido a que o projeto da maioria dos controladores era mais fácil com o filtro mais simples, e este já tinha um desempenho suficiente nas simulações.

A princípio os critério de desempenho desejáveis para uma simulador de voo foram os apresentados na seção 3.1. Contudo, os testes iniciais mostraram que o plataforma não conseguia atender fisicamente todos eles, o maior causa disto era a limitação técnica dos motores usados, tinham uma limitação de velocidade máxima.

Isto afeto a possibilidade do simulador de se movimentar em frequências de até 10 Hz, que seriam usadas para simular turbulências na aeronave. A frequência de funcionamento da aeronave estaria limitada a 1 Hz, o que também afeta o requisito de funcionamento para mudanças causadas pelo piloto, mas ainda se pode considerar para o funcionamento do simulador.

Os outros requisitos se esperam ser atendidos, pois não são muito influenciados pelas limitações do motor Isto foi considerado ao realizar o projeto dos controladores como se explicou na seção 5.4.

Inicialmente será explicado quais os parâmetros dos experimentos realizados, na seção 7.1. Posteriormente serão apresentados os resultados, individualmente para cada controlador projetado e será feita uma análise sucinta do desempenho deles. Na seção 7.2 será apresentado uma análise comparativa do desempenho de todos os controladores.

7.1 Experimentos Realizados

Os experimentos realizados, consistem em fazer a plataforma seguir um referência senoidal no espaço cartesiano, especificamente no eixo **z**.

Embora que, no Sistema de Controle projetado, era possível gerar referências de movimentos em todos os eixos (**x**,**y**,**z**), como se podia observar na figura 72. Isto é devido a que se queria ter uma **avaliação preliminar dos diferentes métodos de controle**, e ainda, se queria obter uma noção sobre as vantagens e desvantagens para realizar e modificar o projeto dos controladores.

Se destaca também, que o uso de referências do tipo degrau para realizar estes testes não é interessante, pois, num cenário de funcionamento normal, o simulador de voo nunca teria um movimento brusco, como o que geraria uma referência do tipo degrau. Embora isto se tenha usado no projeto dos controladores, devido à maior facilidade para realizar e comparar os projetos dos controladores.

Se fizeram três experimentos para cada tipo do controlador, em cada experimento se trocava a amplitude e frequência da senoide de referência no eixo **z**. Cada um deles era projetado para avaliar o desempenho em baixa, média e alta frequência de movimento da plataforma de Stewart, conforme a tabela 4.

Tipo de Senoide	Amplitude	Frequência [rad/s]
Baixa Frequência	20	1.5
média Frequência	10	3
Alta Frequência	7	5

Tabela 4 – Parâmetros da senoide de referência no eixo **z** dos experimentos

Os resultados destes experimentos para cada um controladores projetados serão apresentados a seguir, individualmente para cada método usado para projetar. Estes resultados são apresentados individualmente para cada junta, onde se tem a referência mandada para o controlador, que é igual à referência no eixo **z** depois de passar pela cinemática inversa e o ajuste de referência; junto ao sinal filtrado do potenciômetro, não se usa a medida real, pois esta é muito ruidosa para poder tirar alguma conclusão acerca do controlador.

Também será mostrado o sinal de controle enviado às juntas, em apenas um gráfico, já que só é necessário saber se o sinal do atuador chegou a saturar. De todos estes resultados foi retirado o *offset* de cada sinal, para poder facilitar o entendimento, pois cada junta tem um ajuste um pouco diferente. Por ultimo se ressalta que a gravação de dados foi feita uma vez que o sistema já se encontra em regime permanente.

7.1.1 Experimentos para o Controle PID

Usando os controladores PID projetados com os parâmetros da tabela 3, se obteve os resultados mostrados a seguir na figura 100, para baixa frequência.

Deste resultado se pode observar que a resposta da junta 3 tem uma distorção, isto é causada pelo ruído de medida que não filtrado apropriadamente deste potenciômetro.

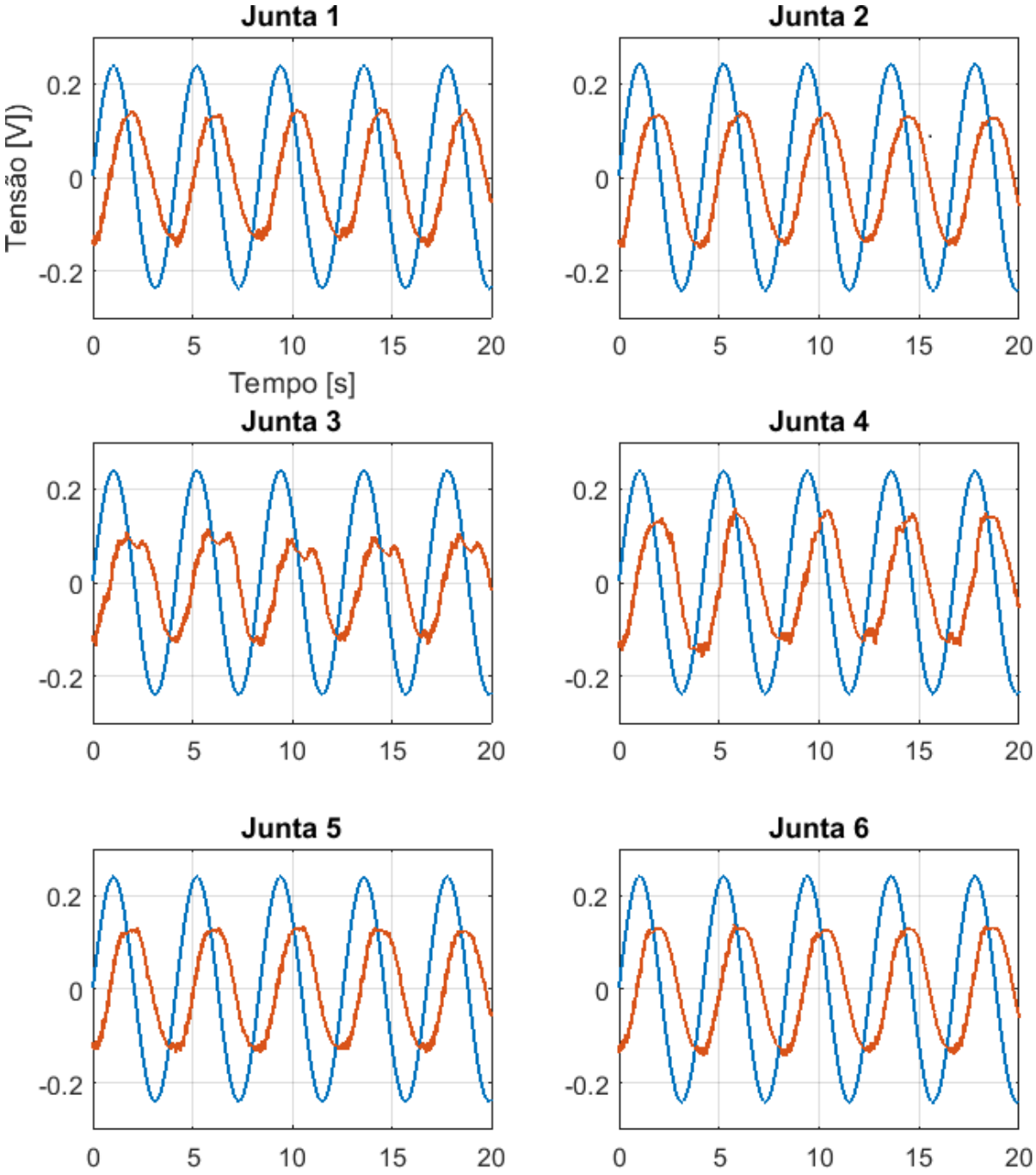


Figura 100 – Resultados para uma referência de baixa frequência com o controlador PID

Existe também uma diferença na amplitude da senoide, isto pode ser explicada por uma provável má calibração do sinal recebido pelo potenciômetro. Ou provavelmente que o sistema em malha fechada tem um ganho menor que 1 para esta dada frequência. A defasagem do sinal é menor a 90°, e 2 s, o que indica que este critério foi cumprido.

Em relação a não ultrapassar os limites físicos do atuador se avalia os sinais de controle enviados pelo controlador. Isto se pode observar na figura 101. Para detectar se existem alguma saturação se pode ver se o valor não ultrapasso a 5 V.

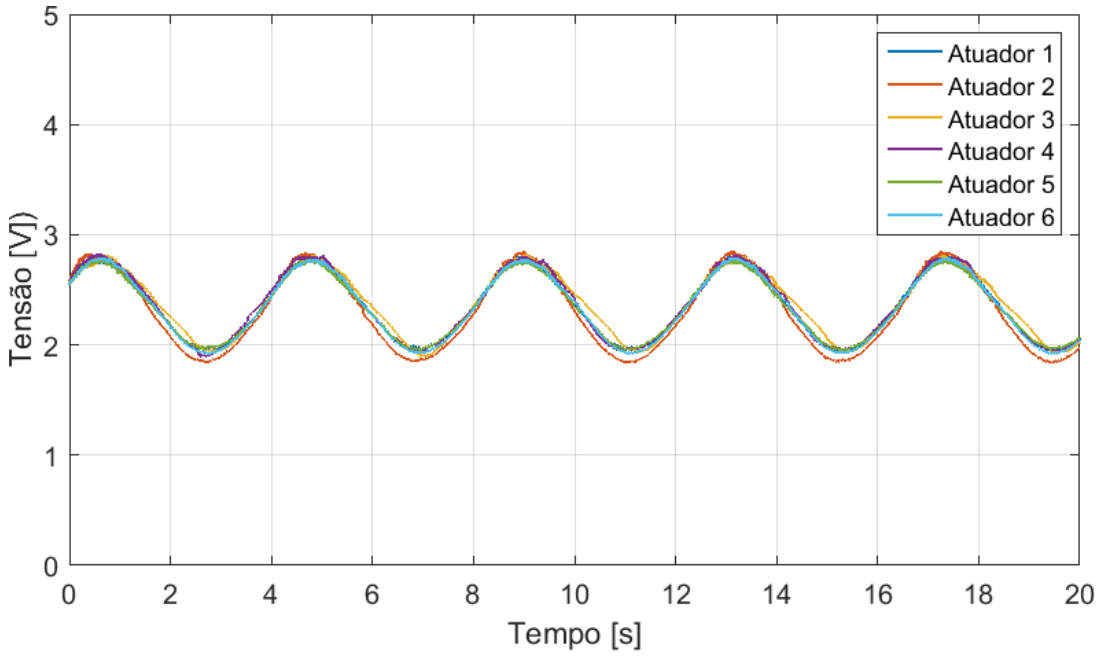


Figura 101 – Sinal de controle para uma referência de baixa frequência com o controlador PID

Os mesmos gráficos serão apresentados para o os outros dois resultados dos experimentos, com frequência média (figuras 102 e 103) e frequência alta (figuras 104 e 105).

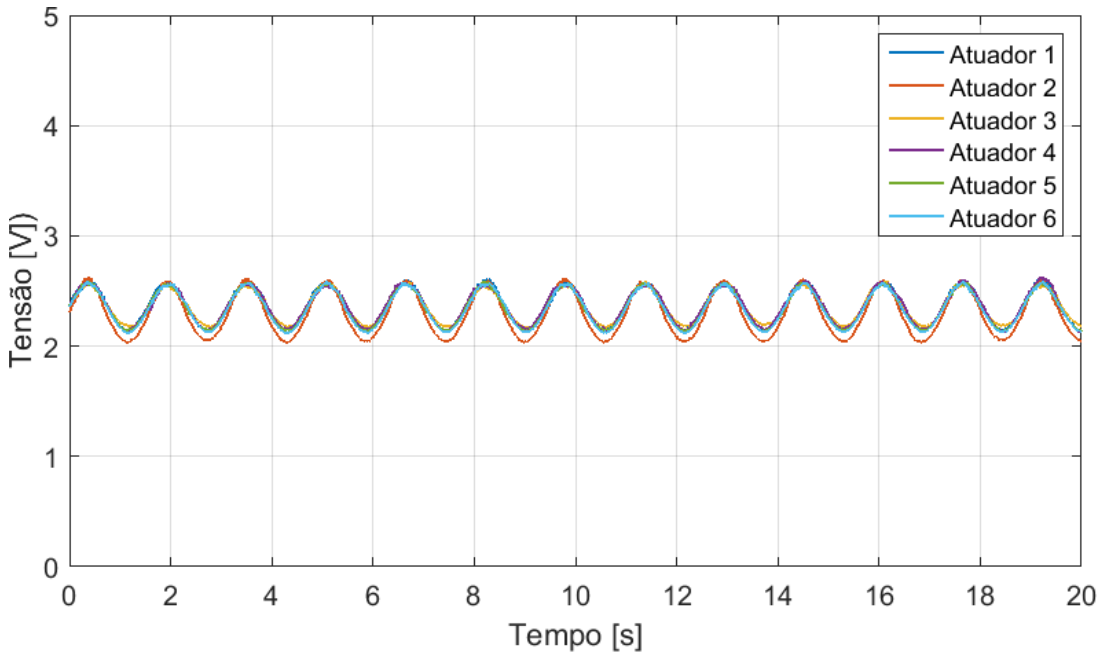


Figura 102 – Sinal de controle para uma referência de alta frequência com o controlador PID

Se observou que para frequências médias e altas o seguimento de referência tem pouco desempenho, com muito ruído presente na medida, ainda que se tenha usado o filtro. Nos testes feitos não se observou que existisse uma defasagem muito notável entre o

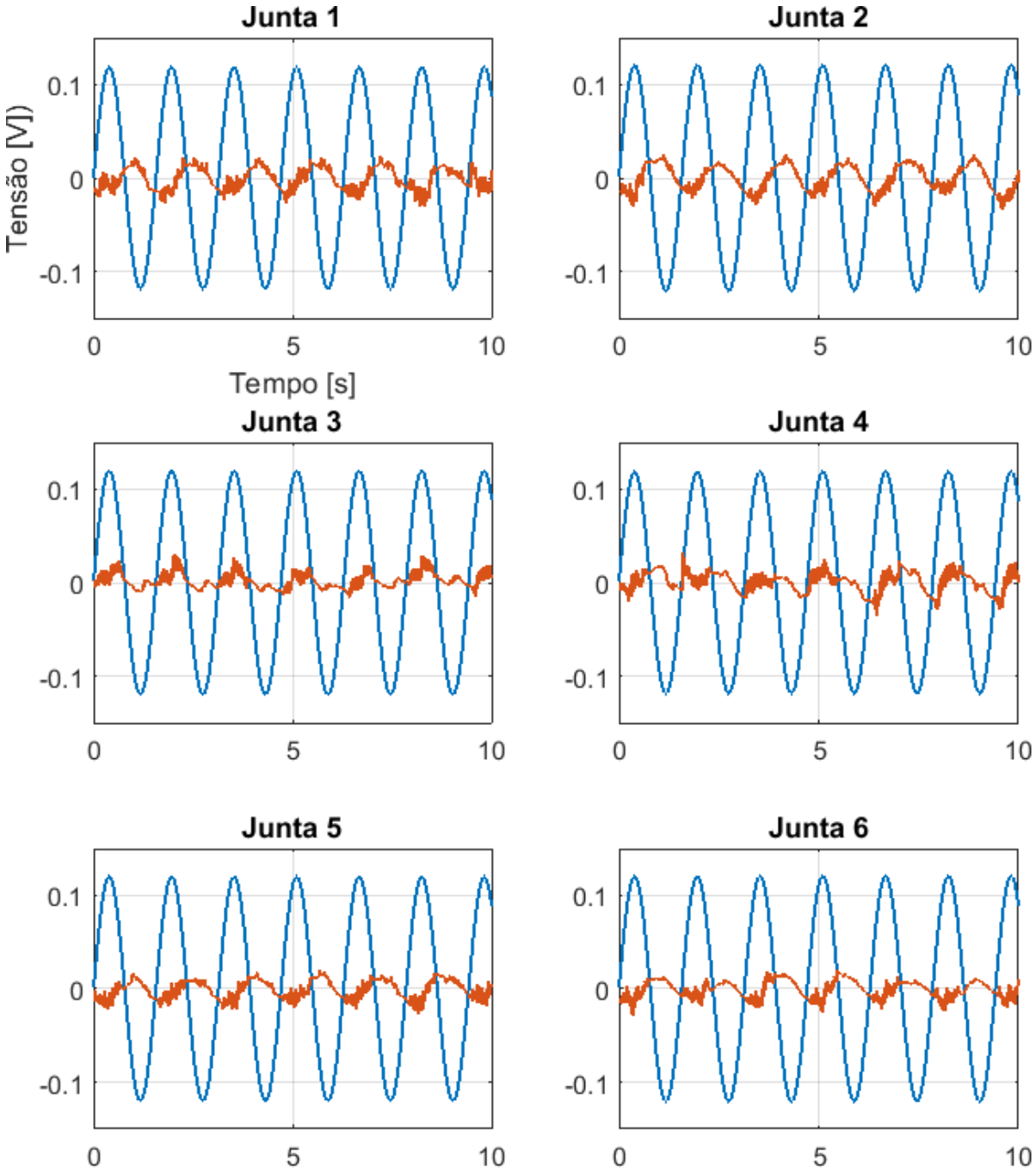


Figura 103 – Resultados para uma referência de alta frequência com o controlador PID

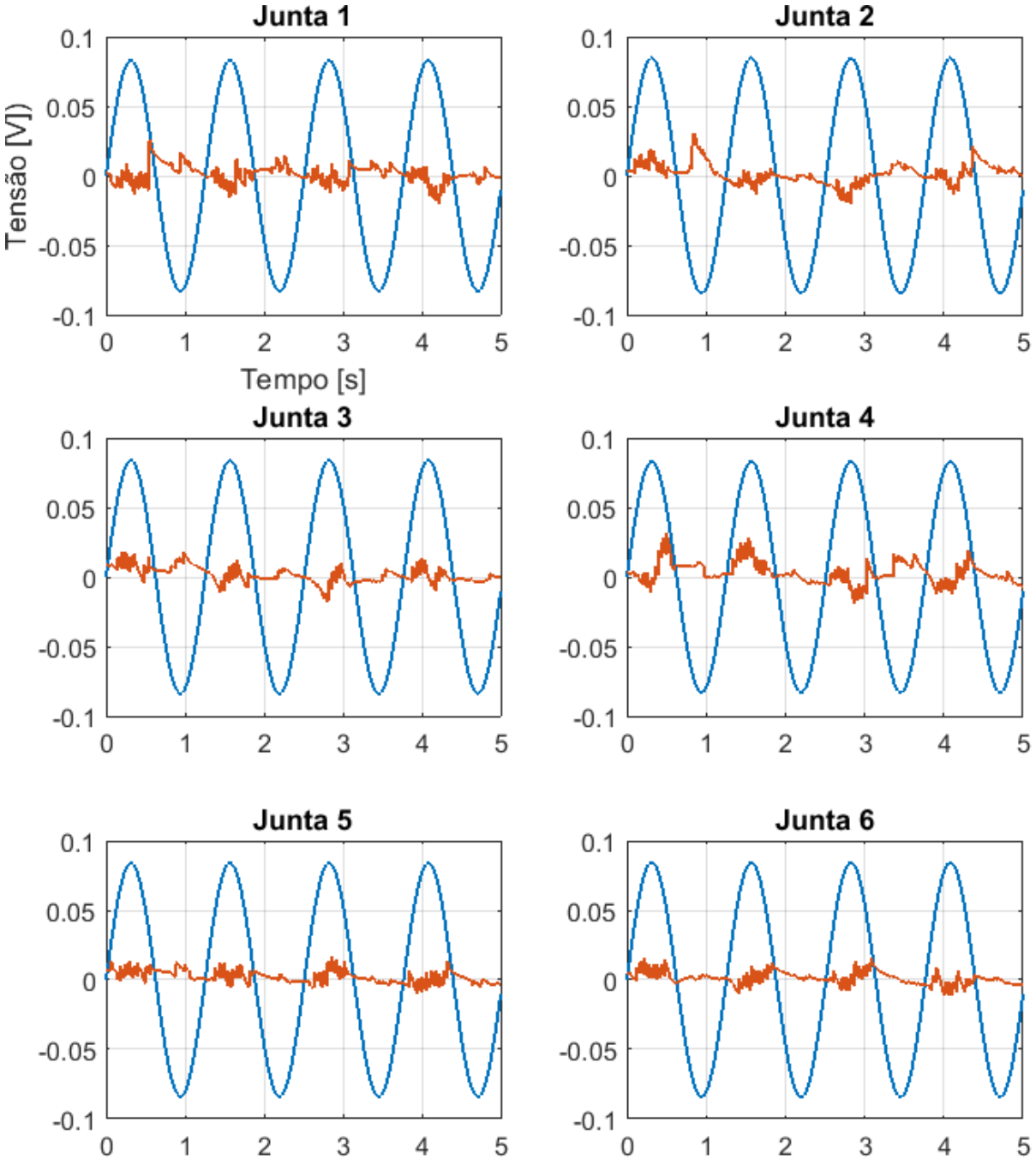


Figura 104 – Resultados para uma referência de média frequência com o controlador PID

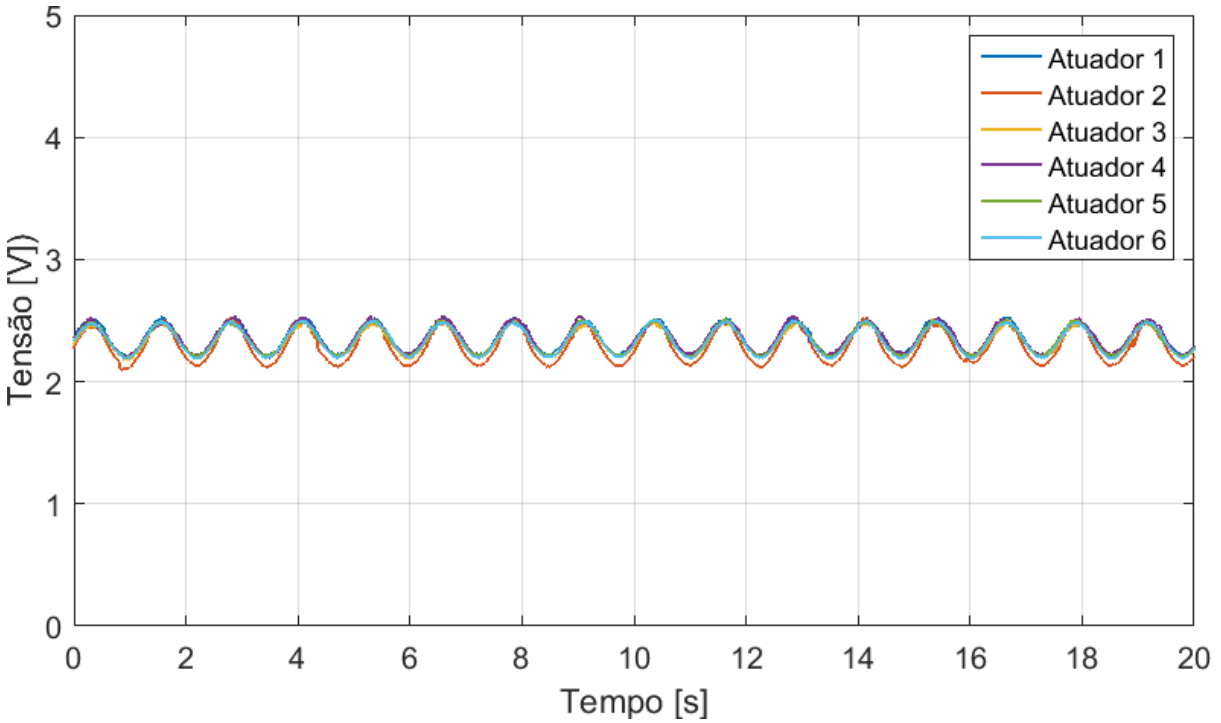


Figura 105 – Sinal de controle para uma referência de média frequência com o controlador PID

acionamento dos atuadores, exceto no caso do atuador 2, que por alguma razão ele tinha uma velocidade um pouco menor que os outros. Se cogita que isto seja por causa que este é o atuador mais velho, ou seja, o que poderia apresentar mais desgaste.

Não se observou em nenhum dos casos que os atuadores tenham saturado, assim poderia simplesmente se identificar um modelo novo para este atuador e sintonizar novamente os controladores PD.

7.1.2 Experimentos para o Lugar das Raízes

Usando os controladores projetados pelo método do lugar das raízes, na seção 5.4.2, definidos pela equação 5.12, se fez os 3 experimentos. Os resultados destes experimentos para a saída filtrada são vistos nas figuras 106, 107 e 108, para baixa, média e alta frequência respectivamente.

Analisando estes resultados se observou que, quanto maior é a frequência, pior é o desempenho deste controlador. No entanto se obteve um melhor resultado para o seguimento de referência de baixa frequência, pois tinha uma menor diferença na amplitude.

O atraso de seguimento para todas as frequências se manteve menor que 2 s, embora a fase, conforme aumentava a frequência, crescia, chegando a quase 180°.

Os sinais de controle respectivos de cada experimento são apresentados nas figuras 109, 110 e 111.

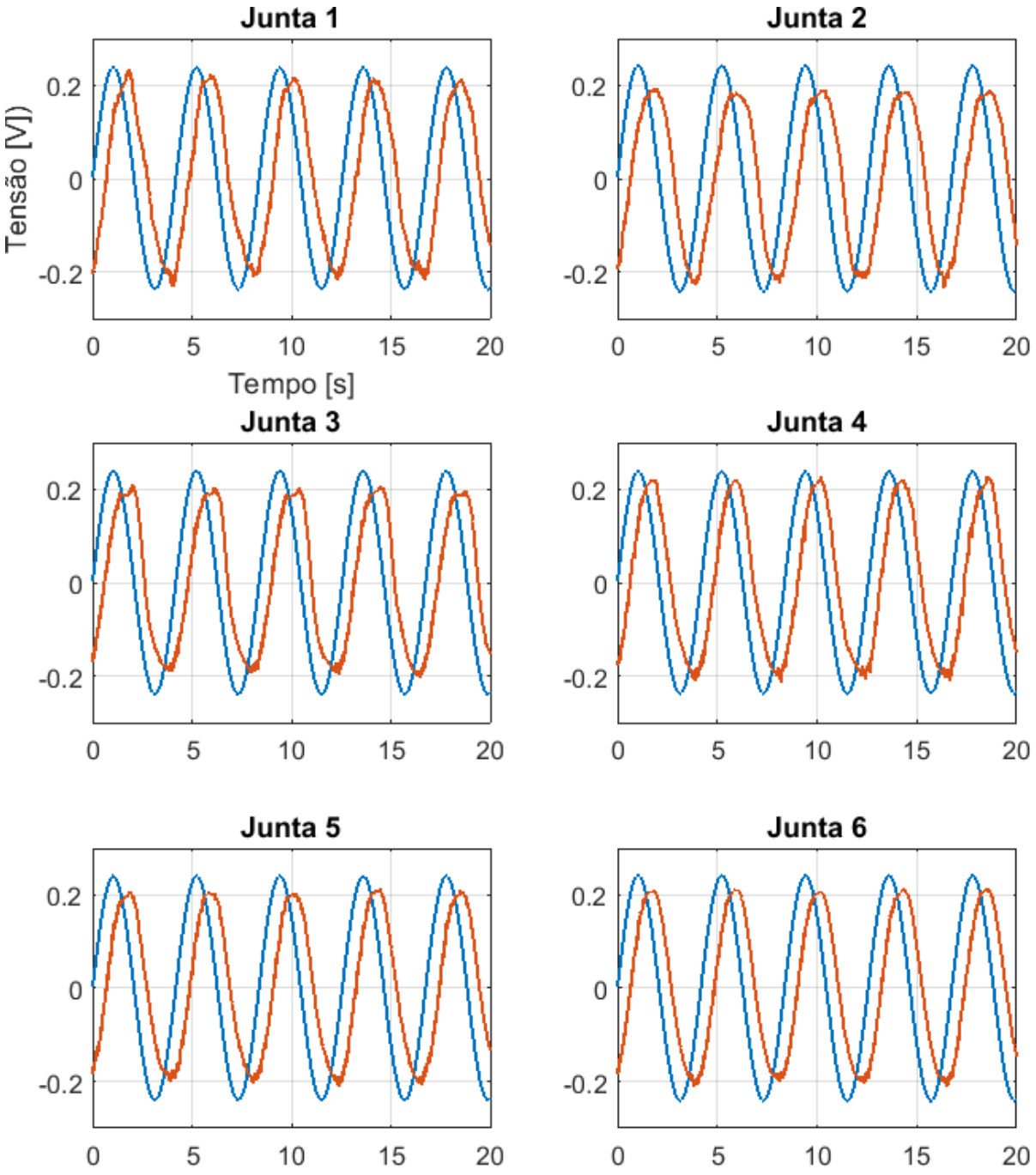


Figura 106 – Resultados para uma referência de baixa frequência com o controlador projeto por metodologia Lugar das Raízes

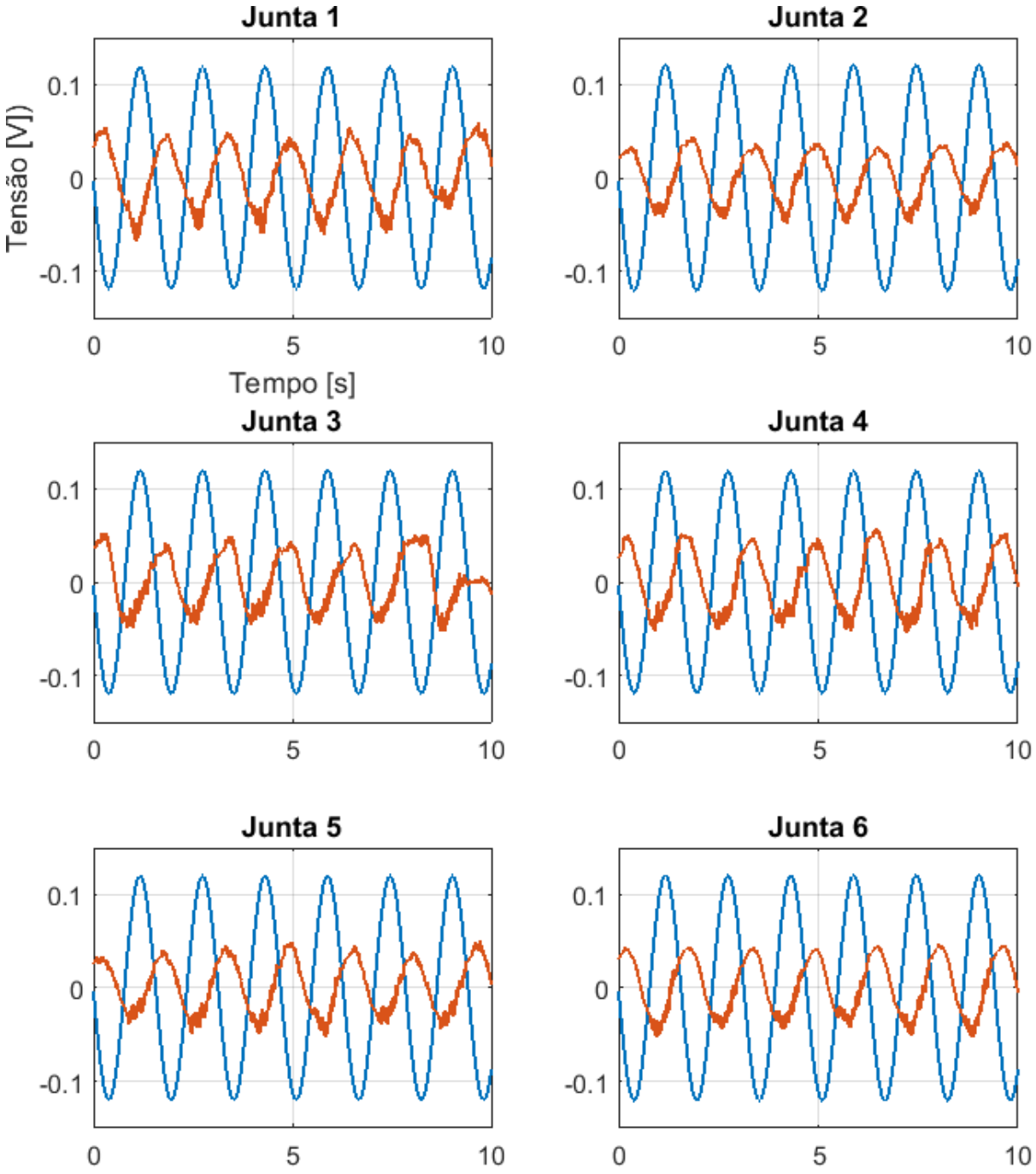


Figura 107 – Resultados para uma referência de média frequência com o controlador projeto por metodologia Lugar das Raízes

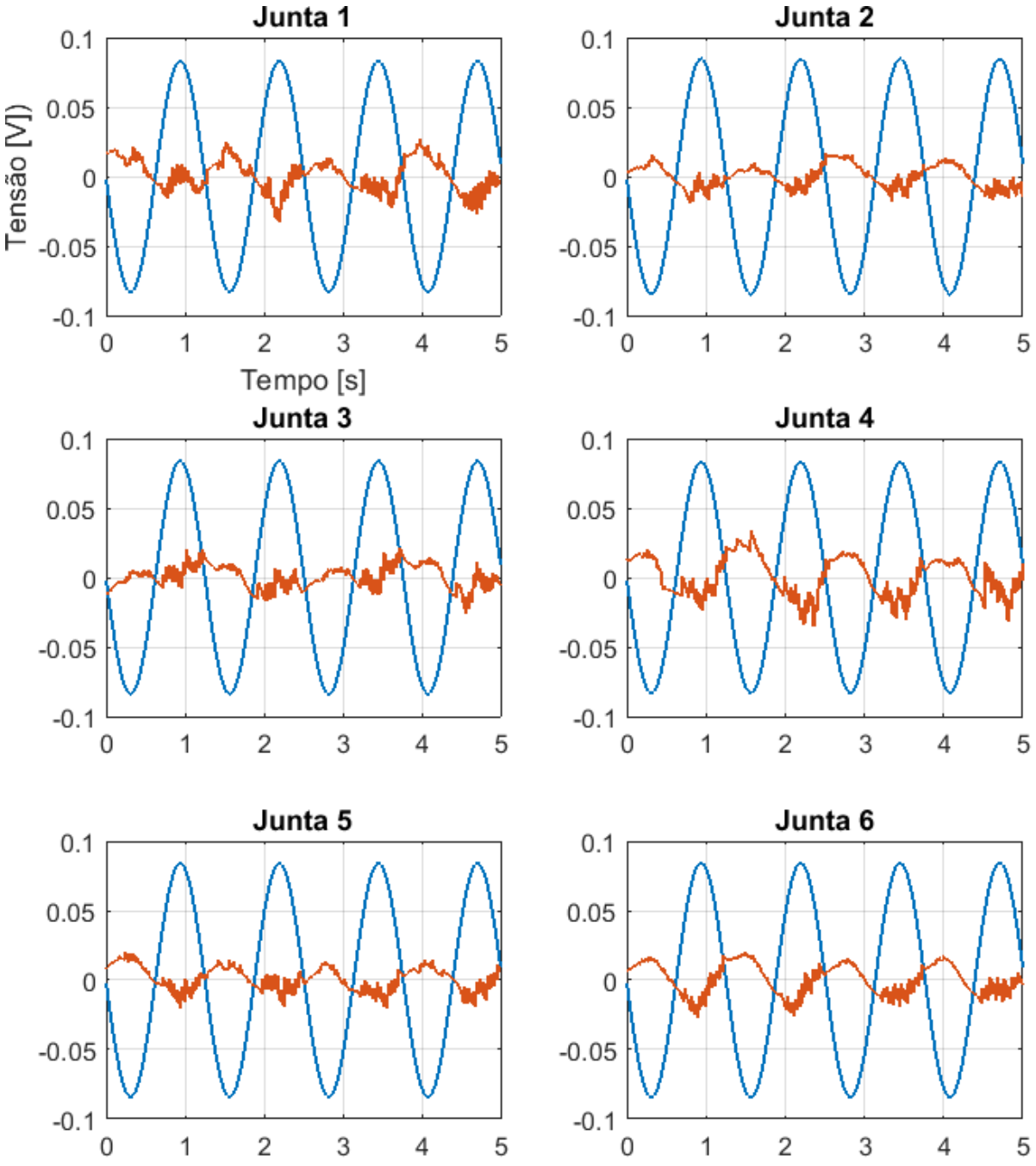


Figura 108 – Resultados para uma referência de alta frequência com o controlador projeto por metodologia Lugar das Raízes

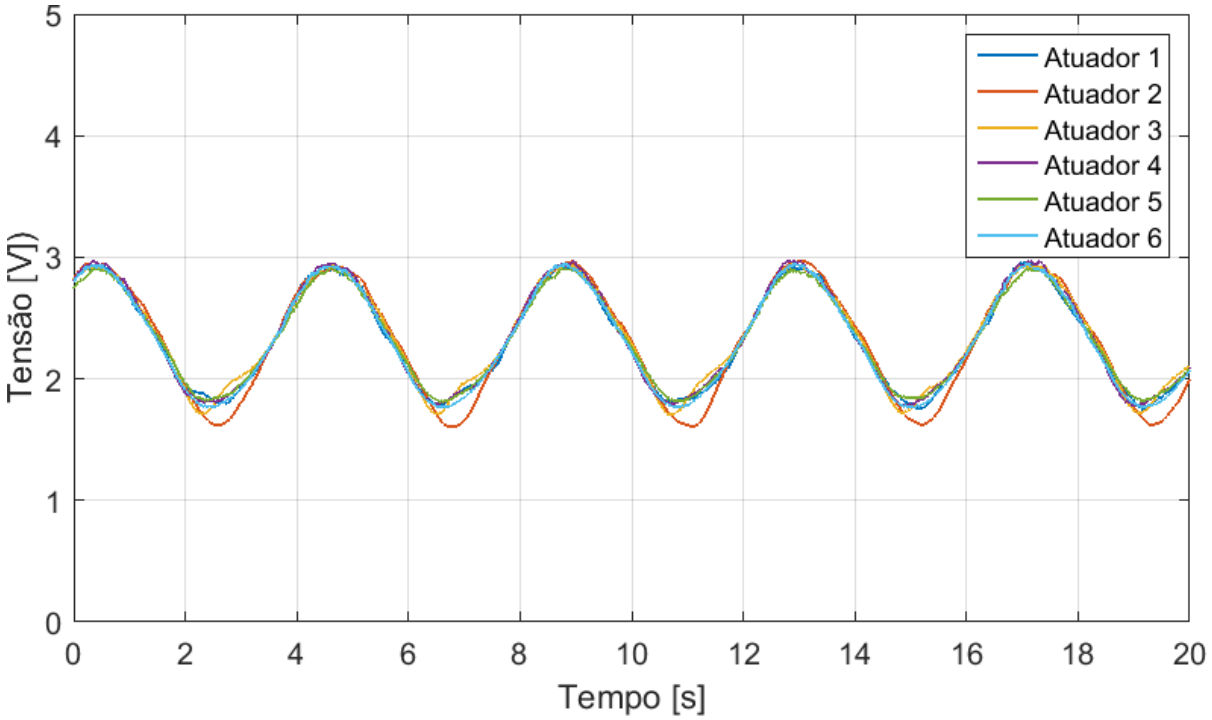


Figura 109 – Sinal de controle para uma referência de baixa frequência com o controlador projeto por metodologia Lugar das Raízes

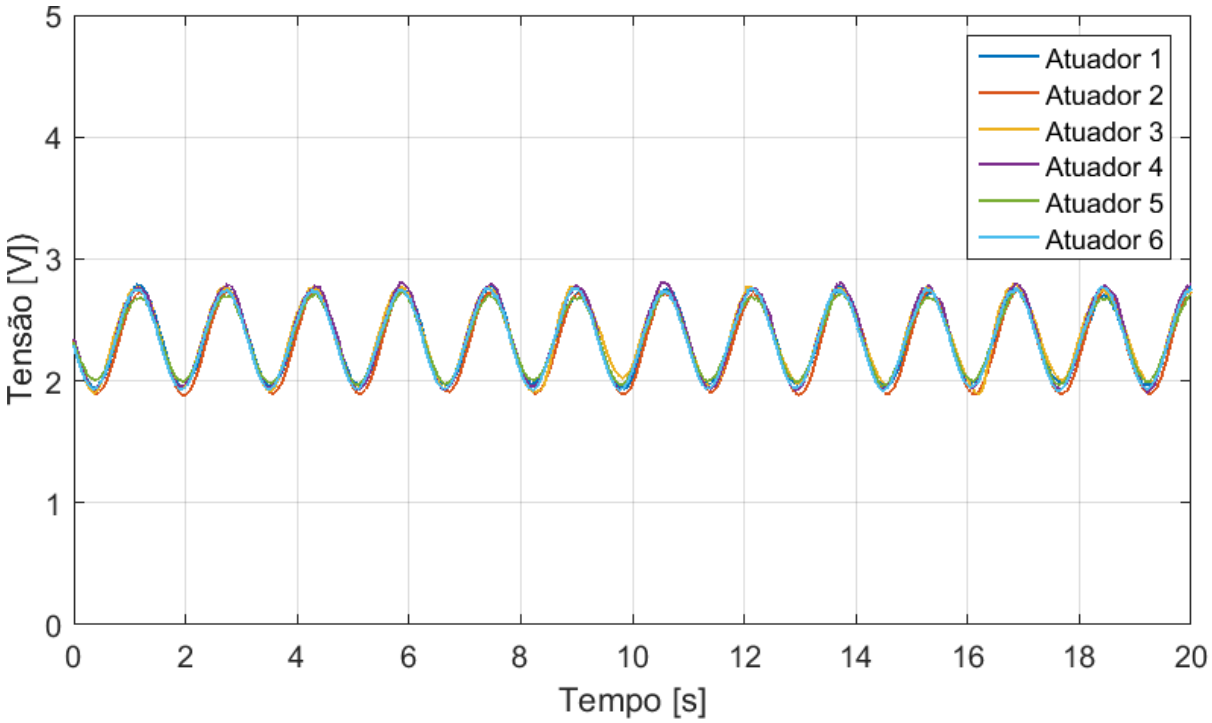


Figura 110 – Sinal de controle para uma referência de média frequência com o controlador projeto por metodologia Lugar das Raízes

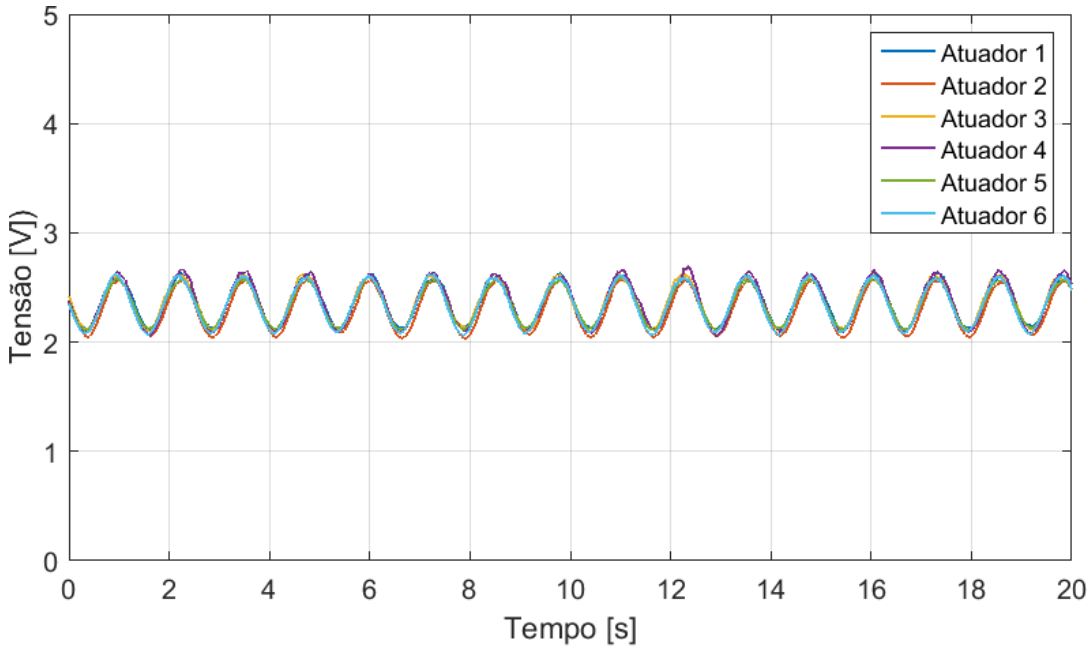


Figura 111 – Sinal de controle para uma referência de alta frequência com o controlador projeto por metodologia Lugar das Raízes

Se pode observar que o limites físicos dos atuadores foram cumpridos e não se tem saturação do sinal de controle enviado aos atuadores.

Se destaca que a ação de controle do atuador 2, para o experimento de baixa frequência (figura 109), apresenta um comportamento diferente quando se encontra em descida, embora isto não acarrete um resultado muito diferente no seguimento da referência.

7.1.3 Experimentos para a Alocação de Polos

O controle por realimentação de estados com controladores projetados pelo método de alocação de polos, na seção 5.4.3, também foi testado na prática e se fez os 3 experimentos.

Os resultados destes experimentos para a saída filtrada são vistos nas figuras 112, 113 e 114, para baixa, média e alta frequência respectivamente.

Analisando estes resultados se observou que, quanto maior é a frequência, pior é o desempenho deste controlador. No entanto se obteve um bom resultado para o seguimento de referência de baixa frequência, pois tinha uma menor diferença na amplitude entre a referência e a saída.

O atraso de seguimento para todas as frequências se manteve menor que 2 s, embora a fase, conforme aumentava a frequência, crescia, chegando pouco menos de 180°.

Os sinais de controle respectivos de cada experimento são apresentados nas figuras 115, 116 e 117.

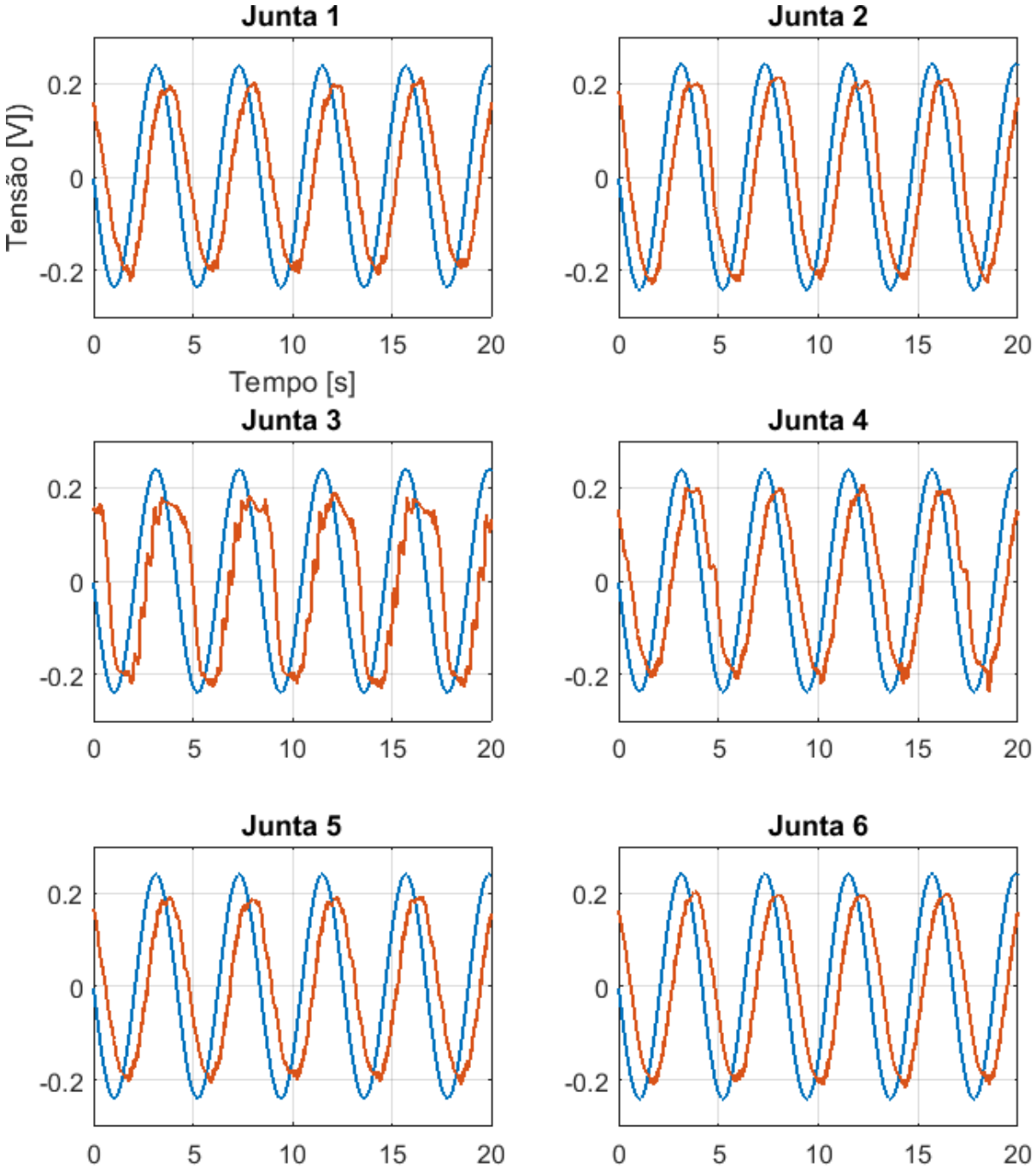


Figura 112 – Resultados para uma referência de baixa frequência com o controlador projeto por metodologia Alocação de polos

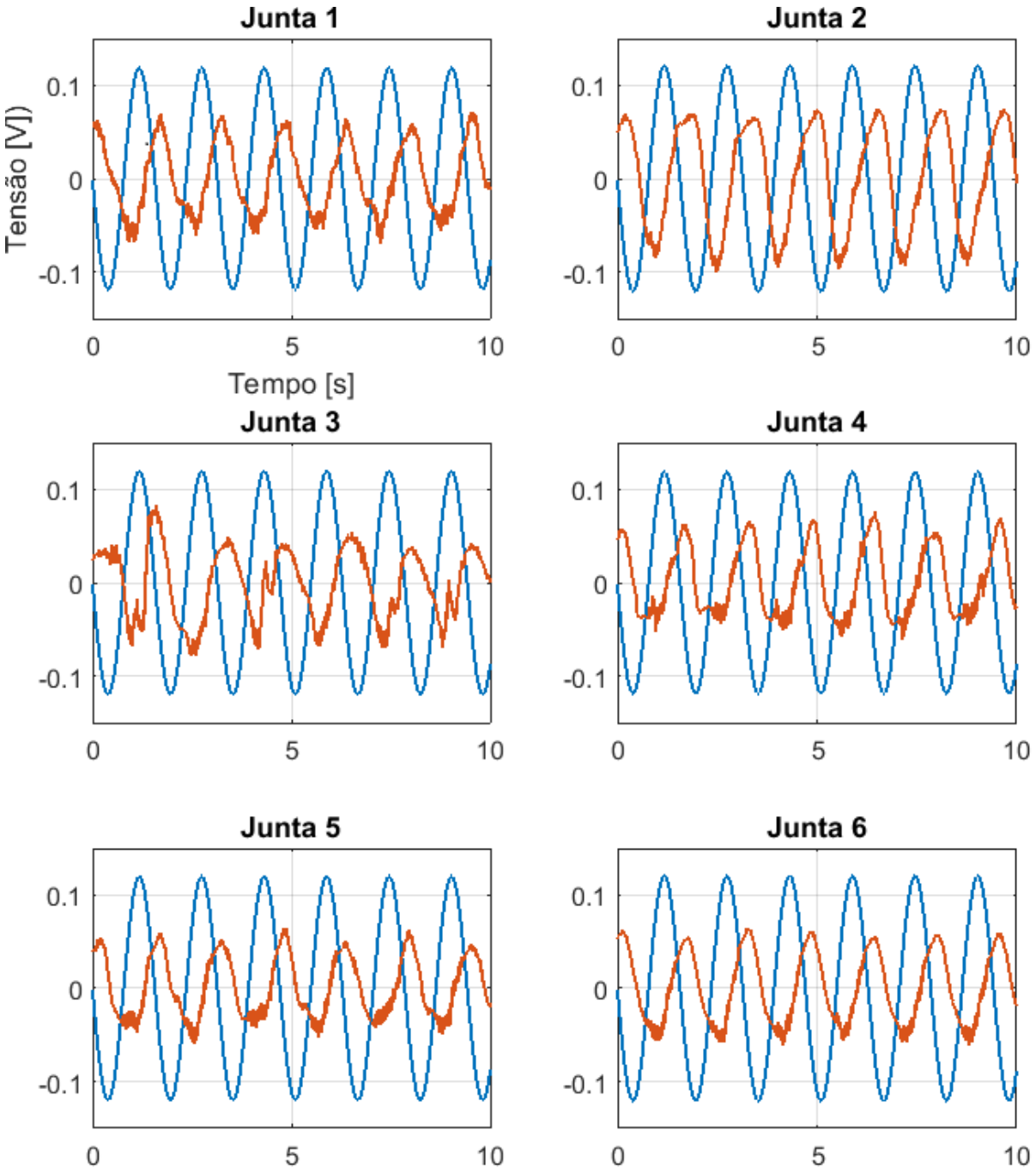


Figura 113 – Resultados para uma referência de média frequência com o controlador projeto por metodologia Alocação de polos

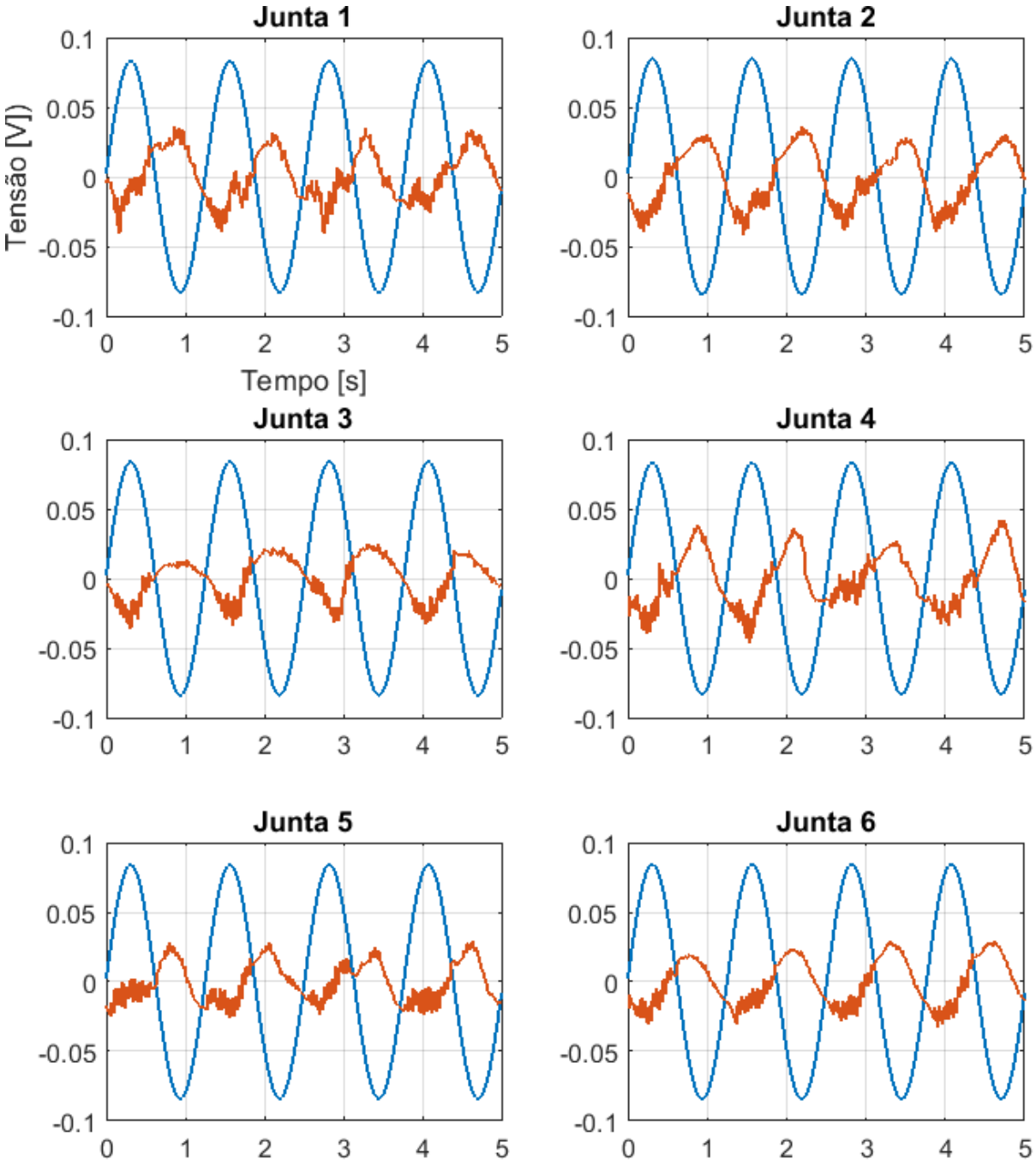


Figura 114 – Resultados para uma referência de alta frequência com o controlador projeto por metodologia Alocação de polos

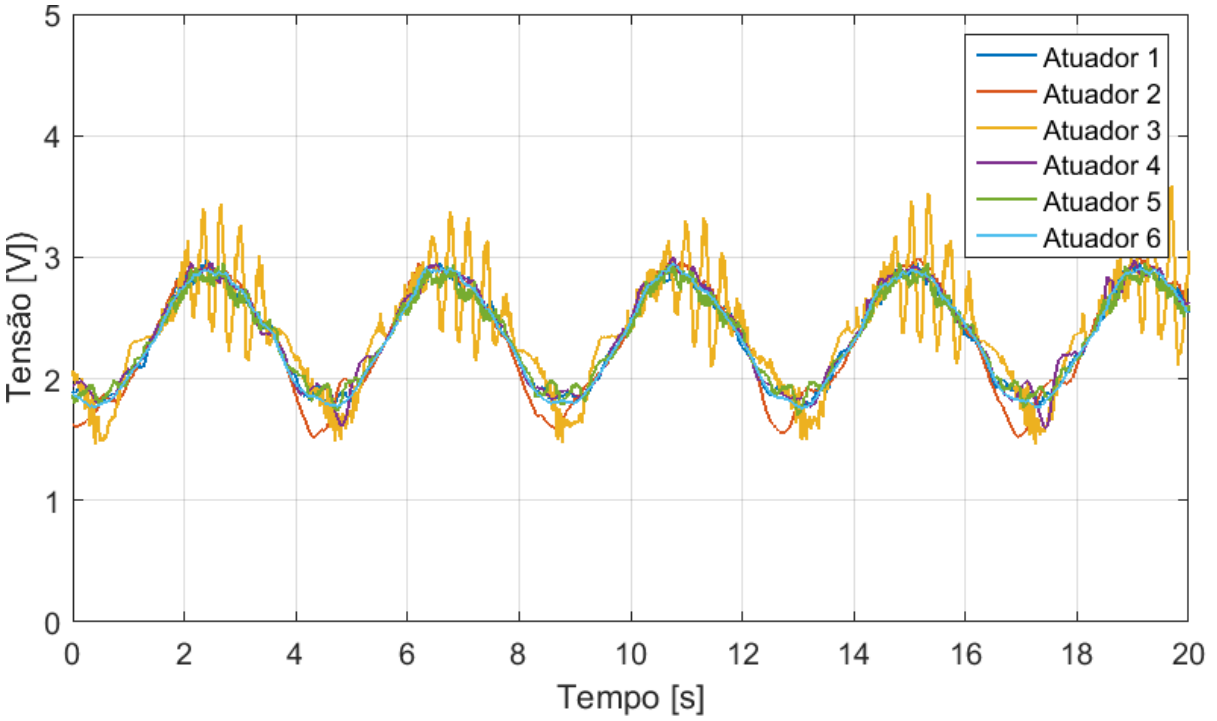


Figura 115 – Sinal de controle para uma referência de baixa frequência com o controlador projeto por metodologia Alocação de polos

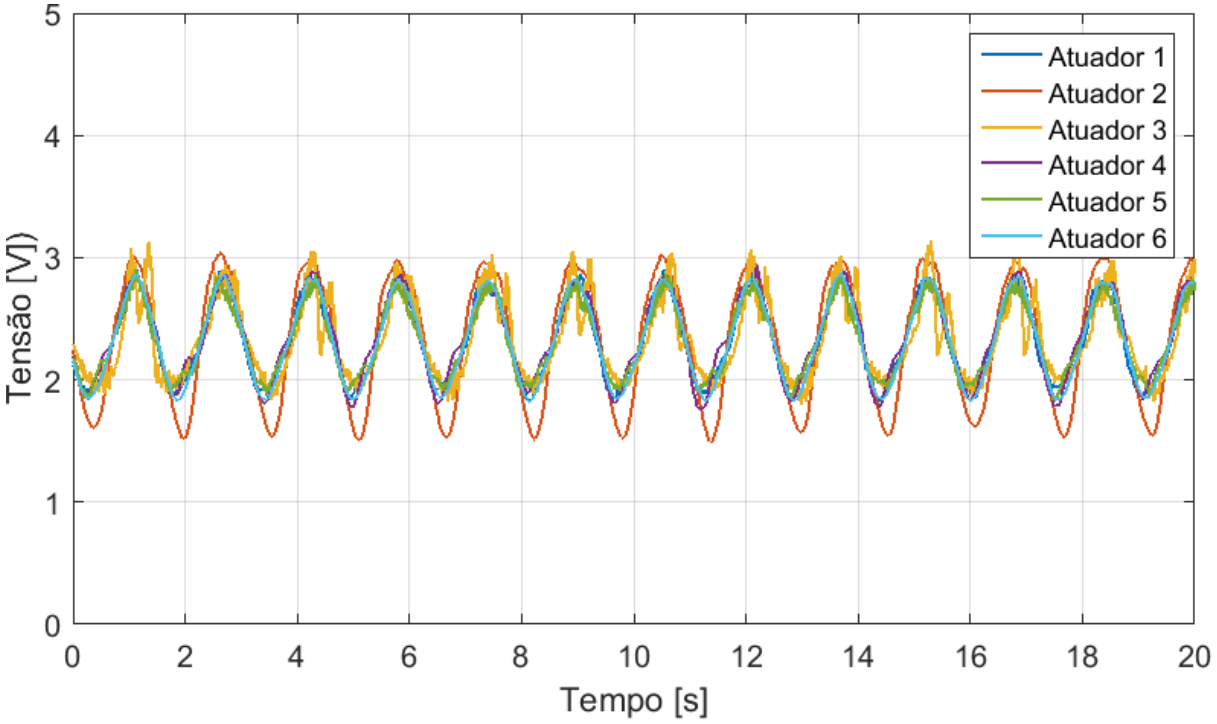


Figura 116 – Sinal de controle para uma referência de média frequência com o controlador projeto por metodologia Alocação de polos

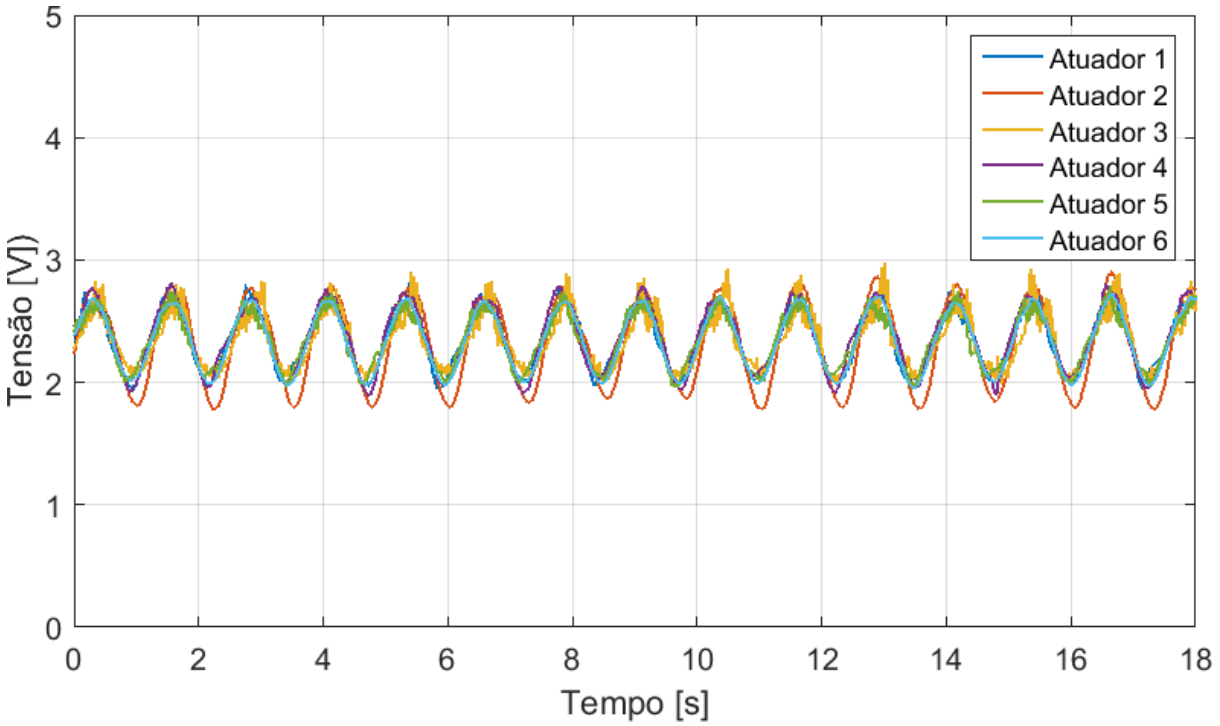


Figura 117 – Sinal de controle para uma referência de alta frequência com o controlador projeto por metodologia Alocação de polos

Se pode observar que o limites físicos dos atuadores foram cumpridos e não se tem saturação do sinal de controle enviado aos atuadores.

No entanto se observa uma oscilação de alta frequência no sinal do atuador 3, especialmente para baixa frequência, isto é causado pelo controle por realimentação de estados, pois esta oscilação era diminuída caso alocasse os polos, do observador deste atuador, mais à esquerda.

Isto se verificou através de testes práticos, e os gráficos apresentados nesta seção usam polos do observador da junta 3, duas vezes mais rápidos que o resto dos observadores. Sem esta mudança a oscilação era maior e causava tremores na plataforma de Stewart. Isto é um problema muito grave, pois pode danificar alguma das juntas não ativas do robô.

Se cogita que a razão destas oscilações reside na interação entre a forma como foi projetado o observador e o ruído, pois o modelo no espaço do estados no qual o observador é baseado trata os estados como derivadas da saída. Saída esta que tem ruído, ainda que seja filtrada, isto quer dizer que quanto maior a ordem do estado estimado, mais será amplificado o ruído que não foi filtrado.

A diferença aparente entre as amplitudes das oscilações para diferentes frequências de referência, se atribui ao próprio comportamento da malha de controle fechada, ou seja, a malha de controle, como um todo, atua como um filtro passa baixa, com uma frequência de corte menor a 5 rad/s,

Se destaca, na ação de controle do atuador 2 para o experimento de média e alta frequência (figura 109), apresenta um comportamento diferente quando se encontra em descida, embora isto não acarrete um resultado muito diferente no seguimento da referência (figura 106).

7.1.4 Experimentos para o Regulador Linear Quadrático

Não se pode obter resultados para o regulador linear quadrático, pois ao se realizar os testes a plataforma não conseguia-se estabilizar, isto é, sem entrada de referência a plataforma começava a vibrar.

A vibração não indicava que o sistema era instável, pois esta não aumentava conforme o tempo, mas tinha um comportamento parecido ao que foi discutido na seção anterior. Ainda que se ajustassem os parâmetros da função de minimização ($\mathbf{Q}$ e $\mathbf{R}$ da equação 3.19), ou os polos do observador.

7.2 Análise Comparativa entre os Controladores

Para se comparar as diferentes metodologias de controle, se levou em conta os requisitos iniciais do projeto. Como já se explicou no começo deste capítulo, os requisitos originais que se tinham para o funcionamento de um simulador padrão, não poderiam ser atendidos, pois os limites físicos dos atuadores não conseguiam atender estes requisitos.

Assim se optou por flexibilizar os requisitos, para se que o Sistema de Movimento operasse até uma frequência de 1 Hz. Mantendo os requisitos de atraso máximo de 2 s para o seguimento da trajetória. Também se deve garantir que todos os atuadores trabalhassem ao mesmo tempo e que não se ultrapassem os limites físicos os mesmos.

Tendo, em conta isto se queria comparar os melhores controladores que se conseguissem projetar usando os quatro métodos descritos nos capítulos 3, 5 e 6. Entre estes apenas o controle LQR, não conseguiu atender os requisitos, pois se tinha problemas graves, que podiam danificar a plataforma, assim este foi descartado.

Também se queria comparar a facilidade com a qual se implementava o este controlador, isto é, se era necessário tomar em conta os polos filtro ou se precisava de um elemento adicional, como um observador; e a facilidade de trocar os parâmetros do controlador, tomando em conta que os outros elementos fossem iguais, por exemplo, mesmo filtro, mesmos polos do observador.

Para facilitar as comparações feitas, se usou o experimento com baixa frequência de todos os controladores testados para um mesmo atuador, o atuador da junta 1, para ilustrá-las. Isto se pode observar na figura 118. Se pode assumir que estas análises também

são validas para o resto das juntas e para os outros testes, com algumas ressalvas que serão discutidas posteriormente.

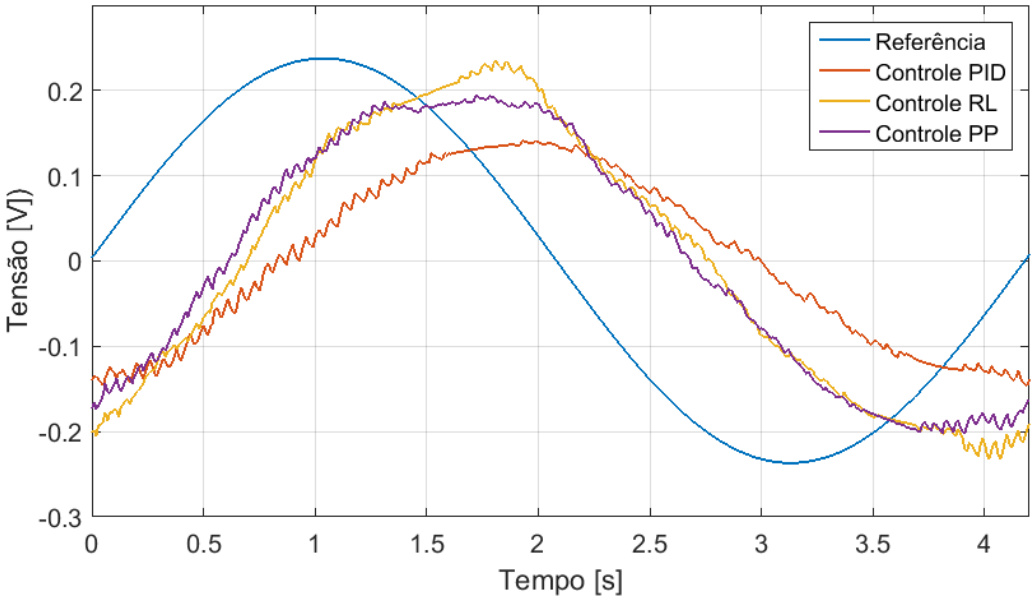


Figura 118 – Comparação entre a saídas das malhas de controle do Atuador 1 com diferentes tipos de controlador, para uma referência de baixa frequência

Se pode concluir que em relação ao atraso presente por causa do filtro e dos diferentes controladores, todos os controladores conseguiam manter o atraso menor de 1 s. No entanto o pior deles era o PID com um atraso aproximado de 0.9 s, enquanto que o resto tinha um atraso aproximado de 0.6 s, sendo que o controle por alocação de polos (Controle PP) tinha uma pequena vantagem.

Em relação a magnitude todos os controladores diminuíam a amplitude da senoide de referência (amplitude aproximada de 0.23 neste exemplo). Neste quesito o melhor resultado o obteve o controle por lugar das raízes (Controle RL), com um amplitude de quase 0.23, mas continua sendo menor que a da referência. O segue o controle por alocação de polos, com uma amplitude um pouco menor, e o pior é o PID com uma amplitude de aproximadamente 0.13, uma diminuição de mais de 40%.

Tomando em conta as observações feitas no momento de projetar o controlador, discutidas já no capítulo 5. Se sabe que o controle mais simples projetados era por sintonização PID, a razão disto era que este não precisava de algum elemento adicional e não era necessário saber quais os polos exatos do filtro. No entanto era complicado mudar o projeto inicial, caso se quisesse melhorar o tempo de resposta, diminuir o ganho estático, etc., pois é difícil fazer com que todos os atuadores tenham o mesmo tempo de resposta.

No caso do método de lugar das raízes, o projeto inicial é complexo, devido a que se tem muitos parâmetros que se podem mudar livremente, outra razão da sua complexidade reside no fato que quanto maior seja a ordem do filtro mais complicado será este projeto,

considerando que os filtros possuem polos relativamente lentos. Este método também torna fazer mudanças no projeto original complicado, pois as vezes não basta apenas mudar o ganho, mas é necessário ajustar as posições dos polos e zeros do controlador. Isto tem que ser feito em cada um dos controladores para que se tenham o mesmo tempo de resposta em todas as juntas, o que aumenta a complexidade.

Para o primeiro método usado para fazer o controle por realimentação de estados, alocação de polos, se considera que o projeto inicial é complicado, pois é indispensável o uso do observador de estados. A ordem do filtro também afeta o projeto deste controlador, tanto quanto o projeto do observador. Contudo este é o método mais simples para realizar mudanças no projeto inicial do controle, pois basta apenas mudar os polos alocados para cada um dos controladores, e ainda como estes polos são os polos de malha fechada se garante que todos as juntas tenham o mesmo tempo de resposta.

No caso do LQR, embora não se tenha conseguido implementar ele satisfatoriamente, se considera que ainda que o projeto dele foi valido. Para ele, a o projeto inicial era mais complicado, pois se tinha que ponderar corretamente as matrizes **Q** e **R**, e ainda era difícil ter uma noção de qual iam ser os polos de malha fechada. Além disto, como o controle também é feito por realimentação de estados, este apresenta as mesmas complicações do método de alocação de polos O mesmo se aplica para mudanças do projeto original que seriam feitas encima das matrizes da função de custo.

A tabela 5 mostra de forma resumida o que foi discutido, onde cada categoria é avaliada numa escala de 1 a 5.

	Complexidade do Projeto Inicial	Complexidade de Alteração de Projeto
PID	**	****
RL	*****	***
PP	****	*
LQR	*****	****

Tabela 5 – Comparação do projeto dos controladores usando metodologias diferentes

É importante destacar que esta é uma análise subjetiva, embora se tenha tentado embasar o raciocínio por trás desta escolha, isto quer dizer que também depende do conhecimento do próprio projetista,

Feito todas as análises se pode concluir que a metodologia de projeto de controle por alocação de polos obteve os melhores resultados entre todas as testadas. No entanto o objetivo inicial do Sistema de Controle do Sistema de Movimento do simulador de voo não pode ser atendido por motivos técnicos e limitações dos motores das juntas prismáticas. Assim se flexibilizaram alguns requisitos deste, e se conseguiu atingir estes novos requisitos.

8 Conclusões e Perspectivas

Neste trabalho se apresentou o projeto realizado para implementar o Sistema de Controle do Sistema de Movimento do simulador de voo que esta sendo construído no GASI. O sistema de controle não conseguiu atender os requisitos originais, devido a limitações dos motores nas juntas prismáticas da plataforma de Stewart. Especificamente, o requisito de funcionamento até 10 Hz não pode ser atendido devido a restrição de velocidade máxima do atuador.

Sendo assim, se optou por flexibilizar este requisito, e a frequência máxima de funcionamento passo a ser 1 Hz. Tendo em conta esta modificação, se estudaram 3 estruturas de controle, estas foram: Controle no espaço das juntas, Controle no espaço cartesiano e Controle baseado na dinâmica inversa. Ainda as duas primeiras estruturas podiam ser centralizadas ou descentralizadas, isto é, o controle podia ser feito integralmente em todas as juntas ou individualmente para cada junta, respectivamente.

Se optou por usar a estrutura de controle descentralizado no espaço das juntas , pois era simples de implementá-la. Nesta estrutura as referências de posição e orientação da plataforma são transformadas em referências de tensão para as juntas ativas da plataforma e se calcula o erro de seguimento por transdutores que medem as alterações das juntas. Para usar esta estrutura foi necessário implementar a resolução da cinemática inversa do robô paralelo. Dentro desta estrutura é possível usar vários métodos para projetar os controladores individuais de cada junta.

Se estudaram e projetaram controladores usando 4 métodos diferentes, estes foram: sintonização PID, lugar das raízes, alocação de polos e LQR. Como a planta a ser controlada, se tratava de uma sistema integrativo, se teve que ajustar os controladores para tomar em conta isto. Assim as duas primeiras metodologias, não possuíam um polo integrativo no controlador, e as duas ultimas, que usam a realimentação de estados, foram modificadas para evitar o uso do integrador.

Por causa da presença de ruído de medida, se planejou usar um filtro de medida. Se estudou e projetou de 4 tipos de filtros passa baixa, eles foram: filtro de 1ª Ordem, filtro Butterworth, filtro Chebyshev Tipo II e filtro Elíptico. Para realizar o projeto se fez uma análise espectral do ruído presente em cada um dos potenciômetros, a fim de obter uma estimativa da frequência de corte que os filtros deveriam ter. Se decidiu usar o filtro de 1ª Ordem, pois os outros filtros tornavam o projeto dos controladores mais complicado, e ainda este filtro possui um desempenho aceitável.

Antes de se poder projetar os controladores, era necessário obter modelos nos qual se basear o projeto deles. Se escolheu por obter estes modelos através de identificação

individual de cada junta a uma entrada degrau. Não era possível usar a identificação usual, pois o sistema tem uma resposta do tipo rampa para um degrau, e, em virtude do ruído de medida, não era concebível usar uma derivada do sinal.

Assim se teorizo uma forma de identificação usando a resposta rampa. O sistema integrativo responde ao degrau como se tivesse sido aplicado uma rampa a um sistema usual. Isto quer dizer que o sistema em regime permanente tende a ter uma derivada estável, ou seja, ou seja o sistema tende a uma assíntota inclinada. Obtendo os parâmetros desta assíntota se consegue aproximar o sistema a um modelo de 2ª Ordem criticamente amortecido, com um polo adicional na origem.

Adicionalmente o uso das técnica de controle moderno requerem o uso de um observador de estados. Se realizaram testes para tentar substituir a ação de filtragem do filtro de medida pela própria dinâmica do observador, mas não se obteve resultados satisfatórios. A identificação, e a validação desta, foram feitas no MATLAB usando dados gravados da plataforma em malha aberta. Todos os controladores foram projetados e simulados usando as ferramentas do MATLAB e do Simulink. Uma vez que se obtinha bons resultados nas simulações, se realizava a implementação para seu uso na plataforma de Stewart. Isto também se aplica para o filtro de medida e a resolução da cinemática direta.

A implementação era realizada inicialmente num modelo do Simulink, usando as bibliotecas da dSPACE, para poder acessar as entradas e saídas do painel I/O, aonde estão conectados os *drivers* dos motores e as medidas dos potenciômetros das juntas. Este modelo foi divido em subsistemas, onde bastava trocar apenas o subsistema dos controladores para realizar mudanças no algoritmo de controle, mantendo o resto intacto. O modelo do Simulink era "compilado" usando o comando `Build Model`, que gerava versões compactadas dele para ser usadas em outros programas. Entre elas se tinha um arquivo com extensão `.sdf` que podia ser usado pelo ControlDesk.

O ControlDesk é um *software* desenvolvido pela dSPACE, com o proposito de implementar sistemas de controle em tempo real. Ele aproveita as capacidade da placa controladora DS1104 que o computador possuía, para garantir que o desempenho do controlador não seja afetado por falta de recursos no PC onde está implementado. Também se tem a capacidade de implementar sistemas supervisórios no ControlDesk, onde se pode controlar e observar todas os parâmetros e variáveis que se tinham no modelo do Simulink original. E ainda se pode gravar e exportar os dados gravados, para serem usados no MATLAB.

Uma vez realizada a implementação, se fez 3 experimentos, com entradas senoidais de diferentes frequências e amplitudes, classificadas em frequência baixa, média e alta, em relação a frequência máxima de 1 Hz, para a referência no eixo **z**. Se coletaram os dados destes experimentos e se realizo uma análise no MATLAB deles, mas também foi realizada

uma análise em tempo real do funcionamento, especialmente para identificar se todas as juntas operavam ao mesmo tempo e com a mesma velocidade.

Finalmente se fez uma análise destes dados para poder comparar o desempenho de cada um dos controladores aplicados. Também se fez uma comparação entre o processo de projetar os controladores, avaliando a complexidade de realizar o projeto inicial deles, e a complexidade de fazer alterações neste projeto original.

Se concluiu que, entre as metodologias de controle usadas, e tomando em conta a dificuldade para projetar estes controles, o controle por realimentação de estados usando a metodologia de alocação de polos, obteve os melhores resultados. Embora que na análise se tenha constatado que o atuador 2 tinha uma velocidade menor que os outros. Se cogita que a razão disto seja o desgaste existente neste motor, que causou que o modelo no qual foi baseado o projeto mudasse, efetivamente mudando o comportamento esperado pelas simulações feitas no Simulink.

8.1 Sugestões para Trabalhos Futuros

Uma das possibilidades para continuação deste trabalho é o uso da IMU disponível, MTi-G-700, para realizar a medição das orientação e posição da plataforma, como forma de verificar com mais precisão se os atuadores estão funcionando com a mesma velocidade. Isto poderia ser observado verificando se existem mudanças na orientação original da IMU ao se realizar uma mudança de posição apenas no eixo **z**.

Poderia se realizar mais experimentos, para analisar o desempenho dos controladores para referências nos outros graus de liberdade da plataforma. Isto também testaria a capacidade dos controladores de compensar o acoplamento existentes entre as juntas, quando se realiza movimentos que não sejam puramente no eixo **z**.

Como uma sugestão de trabalho futuro se poderia realizar o projeto dos controladores usando as outras estruturas de controle apresentadas neste documento, ou usar um controle centralizado. Para isto seria necessário obter um modelo MIMO da plataforma, isto poderia ser realizado usando uma linearização do modelo matemático da plataforma de Stewart que é bastante conhecido na literatura de robótica. [3]

Com este modelo seria possível usar técnicas multivariáveis para projetar os controle. Usando o próprio modelo não linear, se poderia usar metodologias mais complexas, como controle não linear e/ou robusto, como os apresentados em algumas das bibliografias revisadas. [3] Uma outra possível mudança seria de integrar a ação de filtragem com o observador de estados, usando o filtro de Kalman, ou um observador por modos deslizantes.

Uma ultima sugestão seria fazer uma análise na frequência da malha fechada para determinar com mais exatidão o comportamento do sistema para cada frequência para

cada uma das 6 referências existentes no Sistema de Movimento.

8.2 Considerações Finais

Embora se conseguiu atender os requisitos modificados, se considera que os controladores projetados não são necessariamente ótimos. Isto quer dizer que ainda há margem para poder melhorar eles, de forma a atender eles. Também se poderia realizar uma nova identificação do atuador 2.

Lembra-se que o sistema real da plataforma é um sistema não linear, particularmente os motores usados, mas se tentou trabalhar numa região que poderia ser considerada linear, e isto pode ter afetado também o desempenho dos controladores.

O Sistema de Controle não foi implementado unicamente pelo aluno. Se recebeu apoio de outros colegas que trabalham neste projeto e também se obteve apoio do Prof. Mauricio. Algumas dos elementos do sistema de controle foram implementados anteriormente e se reutilizo, ou melhoro, partes que foram desenvolvidas por outros alunos.

Este projeto serviu como uma boa forma de avaliar os conhecimentos do aluno sobre teoria de controle, que foram estudados durante a sua graduação em Engenharia de Controle e Automação na Universidade Federal de Santa Catarina.

Referências

1 STEWART, D. A platform with six degrees of freedom. *Proceedings of the institution of mechanical engineers*, SAGE Publications Sage UK: London, England, v. 180, n. 1, p. 371–386, 1965. Citado 2 vezes nas páginas 9 e 25.

2 BONEV, I. *The True Origins of Parallel Robots*. 2003. Disponível em: <https://www.parallemic.org/Reviews/Review007.html>. Acesso em: 02 nov. 2018. Citado 3 vezes nas páginas 9, 25 e 26.

3 BECERRA VARGAS, M. *Controle de uma plataforma de movimento de um simulador de voo*. Tese de Doutorado em Engenharia Mecânica — Escola de Engenharia de São Carlos - Universidade de São Paulo, São Carlos, São Paulo, 2009. Citado 10 vezes nas páginas 9, 29, 31, 32, 33, 42, 44, 45, 74 e 155.

4 CHROBOTICS LLC. *Understanding Euler Angles*. 2018. Disponível em: <http://www.chrobotics.com/library/understanding-euler-angles>. Acesso em: 02 nov. 2018. Citado 2 vezes nas páginas 9 e 33.

5 POLOLU CORPORATION. *Pololu Jrk USB Motor Controller User's Guide*. [S.l.], 2001. Disponível em: <https://www.pololu.com/docs/0J38>. Acesso em: 06 jun. 2018. Citado 2 vezes nas páginas 9 e 35.

6 DSPACE. *DS1104 R&D Controller Board*. 2018. Disponível em: <https://www.dspace.com/en/pub/home/products/hw/singbord/ds1104.cfm>. Acesso em: 15 ago. 2018. Citado 2 vezes nas páginas 9 e 36.

7 SICILIANO, B. et al. *Robotics: Modelling, Planning and Control*. [S.l.]: Springer Science & Business Media, 2009. Citado 6 vezes nas páginas 9, 41, 45, 46, 47 e 49.

8 OGATA, K.; YANG, Y. *Modern Control Engineering*. 4. ed. New Jersey: Prentice Hall, 2002. Citado 11 vezes nas páginas 9, 41, 50, 51, 52, 53, 54, 55, 68, 69 e 70.

9 MANDAL, M.; ASIF, A. *Continuous and Discrete Time Signals and Systems*. [S.l.]: Cambridge University Press, 2007. Citado 6 vezes nas páginas 9, 58, 59, 60, 61 e 62.

10 CAPPEL K. L. *Motion Simulator*. 1967. US3295224A, 7 dez. 1964, 03 jan. 1967. Citado na página 26.

11 POLOLU CORPORATION. *Datasheet for Glideforce Light-Duty (LD) Linear Actuators*. [S.l.], 2001. Disponível em: <https://www.pololu.com/product/2327/resources>. Acesso em: 05 nov. 2018. Citado 2 vezes nas páginas 32 e 34.

12 DSPACE. *Panels for Single-Board-Hardware*. 2018. Disponível em: <https://www.dspace.com/en/pub/home/products/hw/singbord/conledpanels.cfm>. Acesso em: 15 ago. 2018. Citado na página 36.

13 DSPACE. *ControlDesk*. 2018. Disponível em: <https://www.dspace.com/en/inc/home/products/sw/experimentandvisualization/controldesk.cfm>. Acesso em: 15 ago. 2018. Citado na página 37.

14 XSENS TECHNOLOGIES B.V. *MTi User Manual, MTi 10-series and MTi 100-series*. [S.l.], 2014. Citado na página 39.

15 TANCA NAZAROV, G. C. *Sistema de aquisição de dados em tempo real para avaliação do desempenho de um simulador de voo*. Relatório de Estágio — Universidade Federal de Santa Catarina, Florianópolis, Santa Catarina, 2017. Citado na página 39.

16 FRANKLIN, G. F.; POWELL, J. D.; EMAMI-NAEINI, A. *Sistemas de Controle para Engenharia*. Tradução: Fernando de Oliveira Souza. Revisão técnica: Antonio Pertence Júnior. 6. ed. Porto Alegre: Bookman Editora, 2013. Citado 6 vezes nas páginas 41, 50, 51, 52, 54 e 55.

17 SANTOSO, A. et al. Design and control of the stewart platform robot. In: IEEE. *Modelling & Simulation, 2009. AMS'09. Third Asia International Conference on*. [S.l.], 2009. p. 475–480. Citado na página 41.

18 YINGJIE, L.; WENBAI, Z.; GEXUE, R. Feedback control of a cable-driven gough-stewart platform. *IEEE Transactions on Robotics*, IEEE, v. 22, n. 1, p. 198–202, 2006. Citado na página 41.

19 HUANG, C.-I.; WANG, R.-S. Hybrid control based on type-2 fuzzy-iterative learning control strategies with stewart platform for repetitive trajectories. In: IEEE. *Control Conference (ASCC), 2015 10th Asian*. [S.l.], 2015. p. 1–5. Citado na página 41.

20 YANG, X. et al. Dynamic modeling and decoupled control of a flexible stewart platform for vibration isolation. *Journal of Sound and Vibration*, Elsevier, v. 439, p. 398–412, 2019. Citado na página 41.

21 MATHWORKS. *MathWorks Documentation: buttord*. 2006. Disponível em: <https://www.mathworks.com/help/signal/ref/buttord.html>. Acesso em: 6 jul. 2018. Citado na página 89.

22 MATHWORKS. *MathWorks Documentation: butter*. 2006. Disponível em: <https://www.mathworks.com/help/signal/ref/butter.html>. Acesso em: 6 jul. 2018. Citado na página 89.

23 MATHWORKS. *MathWorks Documentation: cheb2ord*. 2006. Disponível em: <https://www.mathworks.com/help/signal/ref/cheb2ord.html>. Acesso em: 6 jul. 2018. Citado na página 92.

24 MATHWORKS. *MathWorks Documentation: cheby2*. 2006. Disponível em: <https://www.mathworks.com/help/signal/ref/cheby2.html>. Acesso em: 6 jul. 2018. Citado 2 vezes nas páginas 92 e 94.

25 MATHWORKS. *MathWorks Documentation: ellipord*. 2006. Disponível em: <https://www.mathworks.com/help/signal/ref/ellipord.html>. Acesso em: 6 jul. 2018. Citado na página 94.

26 MATHWORKS. *MathWorks Documentation: Control System Designer*. 2015. Disponível em: <https://www.mathworks.com/help/control/ref/controlsystemdesigner-app.html>. Acesso em: 8 ago. 2018. Citado na página 100.