

UNIVERSIDADE FEDERAL DE ITAJUBÁ

PROGRAMA DE PÓS GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

**Análise e Implementação de um Algoritmo de
sensação de Movimento em uma Plataforma de 2
Graus de Liberdade para Treinamento de
Operadores de Caminhões Fora de Estrada**

Lucas Alves Borges

Julho de 2025
Itabira - MG

UNIVERSIDADE FEDERAL DE ITAJUBÁ
PROGRAMA DE PÓS GRADUAÇÃO EM
CIÊNCIA E TECNOLOGIA DA COMPUTAÇÃO

Lucas Alves Borges

**Análise e Implementação de um Algoritmo de
sensação de Movimento em uma Plataforma de 2
Graus de Liberdade para Treinamento de
Operadores de Caminhões Fora de Estrada**

Dissertação submetida ao Programa de Pós-Graduação
em Ciência e Tecnologia da Computação como parte dos
requisitos para obtenção do Título de Mestre em Ciência e
Tecnologia da Computação

Área de Concentração: Matemática da Computação

Orientador: Giovani Bernardes Vitor

Co-Orientador: Fadul Ferrari Rodor

Julho de 2025
Itabira - MG

Agradecimentos

A Deus, minha eterna gratidão, por ter-me concedido saúde, sabedoria e perseverança ao longo de toda esta caminhada acadêmica.

Aos meus orientadores, agradeço profundamente pelo comprometimento, pela orientação, paciência e constante apoio ao longo deste curso. Suas contribuições foram essenciais para meu desenvolvimento acadêmico e para a concretização deste trabalho.

À minha esposa, expresso meu mais sincero apreço pelo apoio inabalável, pela paciência e pela compreensão diante de todas as dificuldades desta jornada. Sua presença foi fundamental para que eu seguisse com equilíbrio, confiança e determinação.

Expresso a mais sincera gratidão aos meus familiares, por terem sido fundamentais na formação do meu caráter e princípios, e aos meus amigos, pelo apoio e pelas contribuições ao longo desta trajetória.

Agradeço especialmente à mineradora Vale S.A., pelo apoio institucional, pelo incentivo contínuo à pesquisa e ao desenvolvimento científico, e pela disponibilização dos recursos necessários, que foram determinantes para a criação deste trabalho.

A cada um de vocês, deixo registrado meu mais profundo e respeitoso reconhecimento.

Resumo

Os simuladores de veículos em ambientes de mineração desempenham um papel cada vez mais vital no desenvolvimento e capacitação de operadores, oferecendo alta segurança, confiabilidade, baixo custo e a possibilidade de criar cenários que seriam difíceis de reproduzir em condições reais. No entanto, devido a restrições espaciais, os simuladores não conseguem replicar integralmente os movimentos dos veículos reais, em especial os caminhões fora de estrada, como o CAT793F, que apresentam comportamentos característicos em aceleração, frenagem, curvas e outros deslocamentos.

Este trabalho apresenta uma metodologia para a implementação de um Algoritmo de *Motion Cueing* (MCA) do tipo *washout* em uma base de movimento com dois graus de liberdade (*degrees of freedom* - DoF), projetada para simular os movimentos de *pitch* e *roll*, condizentes com as condições operacionais dos caminhões em ambientes de mina. O algoritmo proposto foi comparado com a parametrização utilizada por outro autor e apresentou resultados promissores, com melhor representação do movimento, menor atraso de resposta e redução de ruídos provenientes do sistema.

A avaliação dos resultados foi conduzida em três etapas principais: inicialmente por meio da análise objetiva utilizando o OMCT (*Objective Motion Cueing Test*), seguida pela comparação com a resposta esperada do sistema vestibular humano, destacando boa similaridade especialmente nos canais semicirculares. Por fim, foram consideradas as influências de ruídos e perturbações observadas na resposta da plataforma, sendo que esses desvios estão fortemente associados aos efeitos dinâmicos decorrentes da ausência de ancoragem da base ao solo. Essa condição favorece deslocamentos indesejados e desbalanceamentos durante os movimentos, interferindo diretamente na precisão da simulação e na estabilidade do sistema.

Palavras-chave: Algoritmo de sensação de movimento, Plataforma de movimentos, OMCT, Sistema Vestibular.

Abstract

Vehicle simulators in mining environments play an increasingly vital role in the development and training of operators, offering high safety, reliability, low cost, and the ability to create scenarios that would be difficult to reproduce under real conditions. However, due to spatial constraints, simulators cannot fully replicate the movements of real vehicles, especially off-highway trucks like the CAT793F, which exhibit distinct behaviors in acceleration, braking, turning, and other maneuvers.

This work presents a methodology for implementing a washout-type Motion Cueing Algorithm (MCA) on a motion platform with two degrees of freedom (DoF), designed to simulate pitch and roll movements consistent with the operating conditions of mining trucks. The proposed algorithm was compared with an alternative approach and showed promising results, with better motion representation, lower response delay, and reduced system-induced noise.

The evaluation of the results was conducted in three main stages: initially through an objective analysis using the OMCT (Objective Motion Cueing Test), followed by a comparison with the expected response of the human vestibular system, highlighting a good similarity, especially in the semicircular canals. Finally, the influences of noise and disturbances observed in the platform's response were considered, with these deviations being strongly associated with dynamic effects resulting from the lack of anchoring of the base to the ground. This condition favors undesired displacements and imbalances during motion, directly affecting the accuracy of the simulation and the stability of the system.

Keywords: Motion cueing algorithm, Motion platform, OMCT, Vestibular system.

Sumário

	Sumário	6
	Lista de ilustrações	8
	Lista de tabelas	10
1	INTRODUÇÃO	12
1.1	Colaboração Acadêmica na Indústria de Mineração	12
1.2	Plataforma de Movimento e Avaliação de Desempenho	13
1.3	Contribuições	15
2	ESTADO DA ARTE	17
2.1	Utilização de plataformas em simuladores de movimento	17
2.2	Estudo Sobre Algoritmo de Sensação de Movimentos (MCA)	20
2.3	Avaliação de desempenho do MCA	24
3	SISTEMA VESTIBULAR	26
3.1	Canais semicirculares	26
3.2	Órgãos otolíticos	28
3.3	Limiar de sensação de movimento	28
4	PLATAFORMA DE MOVIMENTOS	30
4.1	Cinemática da plataforma de 2DoF	30
4.2	Projeto da plataforma de movimentos	32
5	ALGORITMO DE SENSAÇÃO DE MOVIMENTOS (MCA)	35
5.1	Definição do MCA	35
5.2	Parametrização do MCA	37
6	AVALIAÇÃO DE DESEMPENHO DO ALGORITMO DE SENSAÇÃO DE MOVIMENTO	41
6.1	<i>Objective Motion Cueing Test (OMCT)</i>	41
6.2	Análise da sensação de movimento pelo modelo do sistema vestibular humano	43
6.3	Caracterização do Ruído da plataforma de movimentos	44
7	RESULTADOS E DISCUSSÕES	46
7.1	Validação experimental com o OMCT	46

7.2	Análise gráfica do movimento da plataforma	50
7.3	Avaliação da Resposta Vestibular ao Sinal do MCA	52
7.4	Análise do ruído no algoritmo de sensação de movimento . . .	55
8	CONCLUSÃO	57
	REFERÊNCIAS	59
A	APÊNDICE	63
A.1	Código em Python de funções utilizadas	63
A.2	Código em Python do MCA	65

Lista de ilustrações

Figura 1 – Orientações CAT 793F.	14
Figura 2 – Simulador do caminhão fora de estrada CAT793F.	14
Figura 3 – <i>Antoinette Learning Barrel</i> (ULLRICH, 2022)	18
Figura 4 – Link Trainer (TONG; BIL, 2007)	18
Figura 5 – Plataforma de Stewart (BOHIGAS; MANUBENS; ROS, 2012) . .	19
Figura 6 – Plataforma 2DoF (ZHANG; ZHANG, 2013)	20
Figura 7 – Sistema vestibular humano (PEREIRA, 2021).	26
Figura 8 – Cílio do canal semicircular (BAZONI, 2016).	27
Figura 9 – Movimento do sistema otólito (BAZONI, 2016).	28
Figura 10 – Diagrama 3D da base de movimento utilizada por Zhang e Zhang (2013).	31
Figura 11 – (A) Diagrama esquemático da plataforma de 2DoF. (B) movimento de <i>roll</i> (C) movimento de <i>pitch</i> . Adaptado de Zhang e Zhang (2013)	31
Figura 12 – Plataforma de movimentos com 2DoF desenvolvida. (A) Estrutura real com atuadores e sensores, (B) Vista superior e (C) Vista posterior.	33
Figura 13 – Diagrama de nós e tópicos do ROS	33
Figura 14 – Plataforma de Hardware e comunicação entre os módulos.	34
Figura 15 – MCA clássico, adaptado de Vargas (2009).	36
Figura 16 – Avaliação do valor de ω_n em $G(s)_{HP0}$, adaptado de Vargas (2009)).	37
Figura 17 – Avaliação do valor de ζ em $G(s)_{LP}$	38
Figura 18 – Avaliação do valor de ω_n em $G(s)_{LP}$	39
Figura 19 – Avaliação do valor de ω_n em $G(s)_{HP0}$	39
Figura 20 – Fluxograma do teste OMCT	42
Figura 21 – Fluxograma análise de desempenho do MCA usando modelo Ves- tibular	43
Figura 22 – Fluxograma de simulação em circuito	45
Figura 23 – Resposta em frequência entrada <i>roll</i> saída <i>roll</i>	47
Figura 24 – Resposta em frequência entrada <i>pitch</i> saída <i>pitch</i>	47
Figura 25 – Resposta em frequência entrada X saída <i>pitch</i>	49
Figura 26 – Resposta em frequência entrada Y saída <i>roll</i>	49
Figura 27 – Comparação da simulação com e sem o MCA - <i>roll</i>	51
Figura 28 – Comparação da simulação com e sem o MCA - <i>pitch</i>	51
Figura 29 – Resposta vestibular da rotação <i>Roll</i> para <i>Roll</i>	53
Figura 30 – Resposta vestibular da rotação <i>Pitch</i> para <i>Pitch</i>	53

Figura 31 – Resposta vestibular da translação X para <i>Pitch</i>	54
Figura 32 – Resposta vestibular da translação Y para <i>Roll</i>	55
Figura 33 – Comparação de ruído do sistema com e sem MCA	56

Lista de tabelas

Tabela 1 – Parâmetros do modelo vestibular	25
Tabela 2 – Valores típicos de threshold de percepção para movimentos	29
Tabela 3 – Tabela de parâmetros do MCA	40

Glossário

$\vec{A}_i$	<i>Eixos da base fixa</i>
α	<i>Ângulo de rotação roll</i>
$\vec{B}_i$	<i>Eixos da base móvel</i>
β	<i>Ângulo de rotação pitch</i>
CAN	<i>Controller Area Network - Protocolo de comunicação par sistemas embarcados</i>
CC	<i>Coefficiente de correlação</i>
CWF	<i>Classical Washout Filter - Filtro clássico de sensação de movimento</i>
DoF	<i>degrees of freedom - Graus de liberdade</i>
D_p	<i>Desvio padrão</i>
DTW	<i>Dynamic Time Warping - Distorção dinâmica de tempo</i>
f_a	<i>Aceleração translacional</i>
g	<i>Força gravitacional</i>
MCA	<i>Algoritmo de sensação de movimento (motion cueing algorithm)</i>
MPC	<i>Model Predictive Control - Modelo de controle preditivo</i>
OMCT	<i>Objective Motion Cueing Test</i>
ω_a	<i>Velocidade angular</i>
$\vec{P}_i$	<i>Eixos da base móvel após movimento</i>
PSO	<i>particle swarm optimization - Otimização por enxame de partículas</i>
R	<i>Matriz de transformação de Rotação de movimento linear</i>
RMSE	<i>Erro quadrático médio</i>
ROS2	<i>Robot Operating System 2 - framework de código aberto para desenvolvimento de robó</i>
R_w	<i>Matriz de transformação de Rotação de movimento angular</i>
θ_d	<i>Coordenada de orientação</i>
t_s	<i>Coordenada de translação</i>
τ	<i>Constante de tempo</i>
ΔL	<i>Variação de comprimento dos atuadores</i>
ζ	<i>Coefficiente de amortecimento</i>

1 Introdução

Os simuladores de movimento são sistemas tecnológicos projetados para replicar a experiência física de deslocamento com confiabilidade (STUPANS et al., 2024). Esses dispositivos combinam modelagem matemática, controle de sistemas dinâmicos e a percepção humana para gerar uma sensação realista de movimento em um espaço de trabalho reduzido. Estes simuladores podem ser utilizados em diversas áreas, como na aviação, indústria automotiva e pesquisa. No contexto do treinamento, os simuladores de movimento são amplamente utilizados na formação de pilotos de aeronaves. Esses sistemas permitem a reprodução de cenários realistas sem os riscos e custos associados ao treinamento em equipamentos reais. Segundo trabalhos como os de Grant e Reid (1997), o uso de simuladores contribui significativamente para a melhoria da performance dos pilotos, reduzindo erros operacionais e aumentando a segurança. Apesar disso, pesquisas demonstram que a transferência de habilidades adquiridas em um ambiente simulado para o mundo real não exige a necessidade de um instrutor e de uma experiência física, mas é um recurso essencial para uma aprendizagem segura e eficiente (SANTOS; SILVEIRA, 2019). A indústria automotiva, além de testes automotivos e treinamento de condutores, também emprega esses sistemas para testar novos designs de veículos e validar modelos de dinâmica veicular sem a necessidade de testes físicos extensivos.

Os simuladores de movimento são frequentemente utilizados para estudar a percepção humana ao movimento. Modelos de *Motion Cueing Algorithms* (MCAs) são empregados para gerar estímulos que otimizam a sensação de movimento dentro das limitações físicas da plataforma. O trabalho de Conrad e Schmidt (1970) é uma referência clássica nesse tema, abordando estratégias de filtragem para representar movimentos translacionais e rotacionais de forma perceptivamente correta. Logo, o desenvolvimento e aperfeiçoamento de MCA's permanecem fundamentais para assegurar que os estímulos gerados estejam alinhados com a percepção esperada pelos usuários.

1.1 Colaboração Acadêmica na Indústria de Mineração

A colaboração entre instituições acadêmicas e a indústria tem-se tornado cada vez mais relevante, onde diversas empresas têm investido em parcerias com universidades e outras indústrias com o objetivo de desenvolver sistemas personalizados para atender às necessidades específicas de seus processos. Essas colaborações permitem a criação de soluções voltadas para demandas particulares de cada setor. Este é

o caso da mineradora Vale S.A., que, em parceria com a Universidade Federal de Itajubá - *Campus Itabira*, tem investido no desenvolvimento de um simulador para a capacitação de seus operadores de caminhões fora de estrada em minas a céu aberto.

O treinamento de operadores de caminhões fora de estrada em ambientes minerais apresenta desafios únicos devido às condições adversas e à complexidade da operação dessas máquinas de grande porte. Caminhões como o CAT 793F (CAT,) são utilizados para transportar grandes volumes de material em minas a céu aberto, exigindo habilidades específicas para lidar com terrenos irregulares, cargas pesadas e cenários de risco elevados. A parceria estabelecida entre este projeto e a mineradora Vale tem como foco o desenvolvimento e aprimoramento de simuladores capazes de reproduzir com fidelidade as condições reais de operação. Esses simuladores oferecem uma alternativa segura para a formação de operadores, que podem adquirir experiência prática sem exposição a riscos. Além de promover a segurança operacional (CELESTINO; VALENTE, 2021), essa abordagem contribui para o aumento da eficiência e para a redução de custos no setor mineral, ao diminuir danos aos equipamentos e otimizar processos logísticos.

1.2 Plataforma de Movimento e Avaliação de Desempenho

Os simuladores de movimento também apresentam desafios na construção mecânica e no controle dos atuadores. A plataforma de Stewart é uma das arquiteturas mais utilizadas devido à sua capacidade de gerar movimentos em seis graus de liberdade (*six Degrees of Freedom* - 6DoF). No entanto, plataformas com menos graus de liberdade, como sistemas de 2DoF (Dois Graus de Liberdade - *Two Degrees of Freedom* ou 3DoF (Três Graus de Liberdade - *Three Degrees of Freedom*), também são empregadas em aplicações específicas, como a simulação de veículos terrestres (DELGADO, 2025). O desempenho do simulador depende do design da estrutura, da precisão dos atuadores e da implementação dos algoritmos de controle, sendo que a parametrização adequada do sistema pode afetar significativamente a fidelidade da simulação (GARRETT; BEST, 2010).

O projeto conta com uma parceria estratégica com a mineradora Vale, que forneceu o painel frontal utilizado no simulador. Esse painel é idêntico ao do caminhão fora de estrada modelo CAT 793F, representado na Figura 1, incluindo os comandos associados aos movimentos de *roll*, *pitch* e *yaw*. A utilização de componentes reais contribui para uma experiência de treinamento mais próxima das condições encontradas em campo.

Para realizar a simulação, o algoritmo de sensação de movimento é um elemento fundamental na conversão de movimentos do ambiente virtual em comandos executáveis pela plataforma de movimento, garantindo que as restrições físicas do sistema

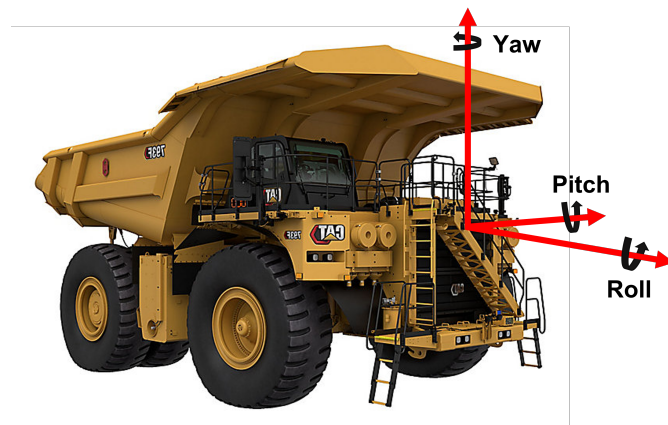


Figura 1 – Orientações CAT 793F.



Figura 2 – Simulador do caminhão fora de estrada CAT793F.

sejam respeitadas. Para isso, o MCA aplica técnicas de filtragem e escalonamento que buscam “enganar” o sistema vestibular humano, responsável pelas sensações de movimento e equilíbrio do corpo, proporcionando uma sensação realista de aceleração e rotação sem a necessidade de replicar fielmente o movimento original. No presente trabalho, o algoritmo foi implementado em uma base de movimento com dois graus de liberdade (2DoF), permitindo a simulação dos movimentos de *pitch* e *roll*, que são essenciais para a reprodução das dinâmicas de operação de veículos de grande porte. A escolha por uma base com 2DoF foi definida para manter uma

relação de compromisso entre desempenho na simulação do movimento e custo de desenvolvimento e manutenção.

Para que simuladores de movimento cumpram seu papel com fidelidade, é essencial que os movimentos gerados representem de forma precisa as sensações esperadas pelos operadores. Nesse contexto, é necessário adotar métodos objetivos que permitam quantificar o desempenho da plataforma em termos de realismo e resposta dinâmica. Entre essas abordagens, destaca-se o teste de indicação de movimento objetivo (*Objective Motion Cueing Test - OMCT*), um procedimento que avalia a qualidade da simulação por meio da análise da resposta em frequência do sistema (SEEHOF; DURAK; DUDA, 2014). O teste baseia-se na aplicação de sinais de entrada bem definidos e na observação da resposta da plataforma, possibilitando a identificação de parâmetros como estabilidade, ganho e fidelidade na reprodução das acelerações.

Além disso, um modelo matemático do sistema vestibular humano é de grande valia para estimar, de forma quantitativa, a percepção sensorial do movimento induzido pelo simulador (RENGIFO et al., 2021). Com essas métricas, é possível avaliar a forma como o movimento é interpretado pelo corpo humano, fornecendo uma métrica mais completa e coerente com a experiência real de operação.

1.3 Contribuições

Este trabalho vai além da análise técnica da simulação de movimento. Ele traz uma série de contribuições relevantes para o desenvolvimento e aprimoramento de sistemas de treinamento baseados em plataformas de movimento, especialmente no contexto da operação de caminhões fora de estrada em áreas de mineração. Entre os principais destaques, podemos citar:

- Aprimoramento do MCA (Motion Cueing Algorithm): A pesquisa contribui diretamente para a compreensão e melhoria do MCA em uma plataforma com 2 graus de liberdade, ajustada às necessidades específicas do cenário de mineração.
- Avaliação em plataforma real e dedicada: O estudo se diferencia ao aplicar o algoritmo em uma base física, analisando o desempenho do sistema de forma prática e realista, dentro de um contexto de uso operacional.
- Utilização em uma base física dedicada de 2DoF para simuladores veiculares: A plataforma utilizada foi desenvolvida especificamente para aplicações veiculares, permitindo investigar sua resposta em condições de operação realistas e reforçando sua aplicabilidade prática.

- Análise de desbalanceamento e ruídos: As características construtivas da base são avaliadas quanto à influência no desbalanceamento de carga e na geração de ruídos indesejados. O trabalho discute estratégias para mitigar esses efeitos, com foco na estabilidade e qualidade da simulação.
- Relevância acadêmica e industrial: Os resultados obtidos são úteis não apenas para o meio acadêmico, como também têm potencial direto de aplicação em simuladores industriais, contribuindo para uma simulação mais realista e uma formação mais eficaz de operadores.

Em síntese, este estudo reforça a importância da integração entre a pesquisa aplicada e as necessidades práticas do setor de mineração, fortalecendo a aproximação entre o meio acadêmico e o setor industrial. Através de parcerias estratégicas, promove avanços na tecnologia de simulação veicular e contribui para a capacitação profissional por meio do desenvolvimento tecnológico orientado a aplicações reais.

2 Estado da arte

Os simuladores de movimento são compostos por três elementos essenciais: o software de simulação, que gera os comandos com base em cenários virtuais; a plataforma de movimento, responsável por executar fisicamente os estímulos; e o algoritmo de sensação de movimento, que adapta esses estímulos às limitações mecânicas da plataforma, buscando preservar a coerência perceptiva. Esta seção abordará os dois últimos elementos, que dizem respeito diretamente ao sistema físico, e, além disso, tratará das abordagens utilizadas no meio acadêmico para a avaliação do desempenho dos simuladores.

2.1 Utilização de plataformas em simuladores de movimento

O desenvolvimento das plataformas de movimento teve origem a partir da necessidade de treinar pilotos em condições seguras e controladas, especialmente durante os períodos de guerra, quando a demanda por formação rápida e eficaz de tripulações era elevada. Um dos primeiros dispositivos criados com esse objetivo foi o *Antoinette Learning Barrel* representado na Figura 3, desenvolvido pela empresa francesa Société Antoinette em 1909 (KRAG, 2017). Esse simulador terrestre consistia em dois tambores sobrepostos com um *cockpit* de pilotagem no topo. O dispositivo era instável nos três eixos e os alunos precisavam manter o equilíbrio da estrutura utilizando os controles da aeronave, desenvolvendo assim os reflexos e a sensibilidade necessários para futuras operações reais de voo. Embora ainda não fosse possível aprender a pilotar completamente, o dispositivo foi um importante marco na criação de simuladores que reproduzem as reações de uma aeronave de forma segura.

A aviação militar do início do século XX exigia meios de instrução que possibilitassem a prática de procedimentos de voo sem os riscos associados ao treinamento em aeronaves reais. Essa necessidade levou à criação dos primeiros simuladores, como o *Link Trainer*, apresentado na Figura 4, desenvolvido por Edwin Link na década de 1930, que se tornou amplamente utilizado pelas forças aliadas durante a Segunda Guerra Mundial. O dispositivo consistia em uma cabine fechada montada sobre um sistema pneumático que permitia simular variações de orientação, respondendo aos comandos do piloto em treinamento.

Nas décadas seguintes, o avanço da engenharia mecânica possibilitou o surgimento de plataformas mais complexas, capazes de reproduzir movimentos em múltiplos eixos. Um dos principais projetos neste ramo foi a criação da Plataforma de Stewart, proposta originalmente por Stewart (1965), em um trabalho desenvolvido para atender



Figura 3 – *Antoinette Learning Barrel* (ULLRICH, 2022)



Figura 4 – Link Trainer (TONG; BIL, 2007)

à demanda da indústria aeronáutica por simuladores de voo para proporcionar treinamento de pilotos com mais segurança e redução de custos operacionais. A arquitetura baseada em seis atuadores lineares dispostos entre uma base fixa e uma plataforma móvel, conectadas por juntas esféricas, permite à estrutura realizar movimentos em seis graus de liberdade, sendo três translacionais e três rotacionais. Essa configuração proporciona alta rigidez e precisão na reprodução de manobras complexas, tornando-se o padrão mais utilizado em simuladores até os dias atuais (ZHAO et al., 2025). Além de sua ampla aplicação na aviação, a Plataforma de Stewart também é empregada em diversos campos, como cirurgias médicas, astronomia, entretenimento e pesquisa científica (ALINA-ELENA; GHEORGHE, 2023). Sua estrutura é amplamente estudada, o que contribui para o domínio da cinemática inversa em aplicações práticas. Já a cinemática direta, por outro lado, apresenta maior

complexidade, exigindo abordagens numéricas mais sofisticadas para sua resolução (KARMAKAR; TURNER, 2023).

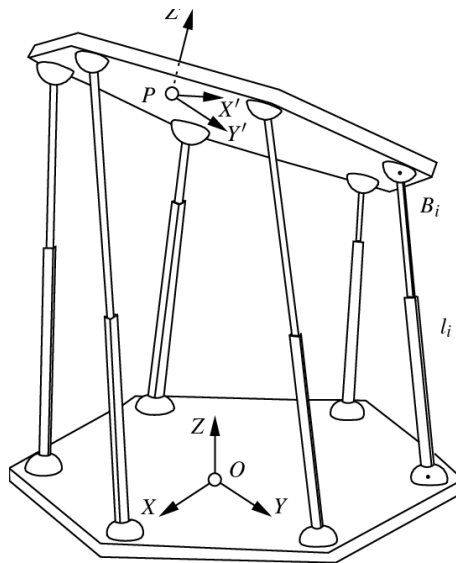


Figura 5 – Plataforma de Stewart (BOHIGAS; MANUBENS; ROS, 2012)

A consolidação dessa arquitetura abriu espaço para inovações na geração de movimento percebido. Inicialmente controladas por circuitos analógicos, filtros de "lavagem de sinal" (*washout*) eram usados na década de 70 para filtrações em tempo real de sinais inerciais complexos. Esse processo evoluiu mais adiante para meios digitais, culminando nos algoritmos de sensação de movimento (Motion Cueing Algorithms - MCA), responsáveis por transformar perfis de aceleração e orientação em comandos que, mesmo limitados pelo curso e velocidade dos atuadores, induzem sensações fidedignas no usuário.

Por outro lado, para simuladores de veículos terrestres as variações de aceleração se diferem dos aviões, sendo a frequência de movimentos dos veículos terrestres mais intensa e constante (NEHAOUA et al., 2006). Contudo, para estes veículos a variabilidade de movimentos é mais reduzida, desta forma, plataformas com apenas dois graus de liberdade têm-se mostrado suficientes para representar, de forma eficaz, as sensações de rotação e também de translação (ARIOUI; NEHAOUA; AMOURI, 2005). Isso é possível graças aos algoritmos de sensação de movimento que são capazes de simular acelerações translacionais com movimentos de inclinação *roll* e *pitch*.

Um protótipo de simulador de movimento com dois graus de liberdade (2DoF) foi desenvolvido por (LIU; LI; ZHAO, 2023), utilizando atuadores pneumáticos para reproduzir acelerações lineares e velocidades angulares de veículos. O estudo demonstra que plataformas 2DoF podem ser uma alternativa eficiente e de baixo custo frente às tradicionais plataformas de 6DoF, especialmente quando combinadas com algoritmos adequados de controle e processamento de sinais. Ao utilizar um

algoritmo de fusão de dados, o autor obteve um sistema que proporciona respostas suaves e coerentes com a percepção humana, apesar de ligeiramente menos preciso.



Figura 6 – Plataforma 2DoF (ZHANG; ZHANG, 2013)

A base de movimento utilizada no presente trabalho é bastante semelhante à proposta por (ZHANG; ZHANG, 2013), que apresenta um manipulador paralelo 2DoF com duas pernas lineares e uma de apoio, projetado para aplicações em simuladores de direção, como pode ser visto na Figura 6. Essa configuração permite evitar interferências entre os atuadores e a plataforma, além de oferecer uma estrutura simples e de baixo custo. A plataforma é capaz de gerar movimentos de *pitch*, *roll* e vibração, atendendo aos requisitos para sensação de movimento por meio do simulador. A viabilidade do seu sistema foi demonstrada por meio de simulações e da construção de um protótipo funcional, evidenciando seu potencial para aplicações práticas em simuladores veiculares, assim como pode ser estendida para simulador de caminhão fora de estrada.

2.2 Estudo Sobre Algoritmo de Sensação de Movimentos (MCA)

Impulsionados pela necessidade de criar um método seguro e eficaz para o treinamento de pilotos, especialmente no contexto da aviação nas décadas de 1950 e 1960, surgiram os primeiros esforços para o desenvolvimento de sistemas capazes de simular, de forma realista, as sensações físicas da pilotagem. Diante das limitações mecânicas das plataformas de movimento, tornou-se essencial o desenvolvimento de algoritmos capazes de converter os movimentos desejados da simulação em comandos fisicamente viáveis. Foi nesse cenário que surgiu o primeiro modelo de Algoritmo de Sensação de Movimento (*Motion Cueing Algorithm* – MCA), proposto por Conrad e

Schmidt (1970), que se consolidou como base para os simuladores dinâmicos modernos. O MCA é um componente essencial para a reprodução de sensações de movimento, tendo como objetivo transformar os sinais de entrada da simulação em comandos adequados para os atuadores da plataforma, garantindo que a experiência do operador seja realista e compatível com as capacidades físicas do sistema. Para isso, aplica filtros e técnicas de compensação, removendo componentes indesejados e otimizando os comandos enviados à plataforma, de modo que o operador perceba acelerações e mudanças de atitude semelhantes às encontradas em cenários reais, sem comprometer a simulação (STUPANS et al., 2024).

Utilizando a técnica clássica de sensação de movimento, Fuentes (2021) implementou um algoritmo em uma plataforma com seis graus de liberdade, com o intuito de reproduzir sensações inerciais realistas em simuladores de direção. O estudo foi estruturado em duas etapas: a primeira consistiu em uma revisão teórica aprofundada sobre simuladores de alta fidelidade e os diversos MCAs presentes na literatura, a segunda envolveu a aplicação prática do algoritmo em ambiente de simulação, empregando dados de um videogame para simular o comportamento veicular. A validação do modelo foi realizada por meio da comparação entre a resposta percebida pelo usuário no simulador e a resposta esperada no veículo real, demonstrando resultados coerentes e promissores.

A partir da estrutura proposta pelo MCA clássico, diversos modelos alternativos foram desenvolvidos ao longo dos anos com o objetivo de superar suas limitações e melhorar a fidelidade das sensações de movimento. Como o MCA clássico não considera diretamente um modelo matemático da percepção humana como forma de orientar o ajuste dos parâmetros, foi proposto por Asadi et al. (2016) um filtro *washout* no qual foi aplicada a otimização por enxame de partículas (*Particle Swarm Optimization* - PSO) para esse fim. O PSO foi empregado para automatizar o processo de sintonia dos parâmetros do filtro, substituindo o método empírico tradicional por uma abordagem baseada em critérios quantitativos. O modelo considera aspectos como as limitações físicas da plataforma, o erro de sensação entre motoristas reais e simulados e os efeitos dos limiares humanos na coordenação por inclinação. No entanto, um desafio importante associado ao uso de modelos adaptativos nesse contexto está na definição da função de custo, uma vez que mensurar com precisão a discrepância entre sensação real e simulada exige um modelo sensorial confiável e de difícil construção. A implementação foi realizada em simulação e os resultados obtidos pelo autor indicaram melhorias no desempenho do MCA e principalmente na eficiência do uso do espaço de trabalho.

Seguindo o mesmo princípio, outros modelos de MCA adaptativos vêm sendo investigados na literatura. Métodos baseados em redes neurais, por exemplo, têm sido propostos com o objetivo de melhorar a fidelidade das sensações de movimento

por meio da adaptação a diferentes faixas de condução, como velocidades baixas, médias e altas (QAZANI et al., 2023). Outro exemplo é a utilização de algoritmos genéticos (GA) aplicados por Asadi et al. (2015) para otimizar parâmetros dos filtros *washout* não lineares com base em critérios como percepção humana, limitações físicas e coordenação por inclinação. Com isso, observam-se maior precisão dos movimento gerados, o que contribui para um uso mais eficiente da faixa operacional da plataforma. Apesar dos avanços, em ambos os métodos, estes ainda enfrentam desafios quanto à sua viabilidade em aplicações dinâmicas em tempo real, devido à complexidade computacional envolvida nesses processos de otimização.

Apesar da ampla utilização dos filtros clássicos de *washout* em simuladores de movimento, o Controle Preditivo Baseado em Modelo (MPC) tem se destacado como uma alternativa promissora por sua capacidade de incorporar restrições físicas e antecipar o comportamento do sistema ao longo de um horizonte de previsão. Com isso, o MPC consegue otimizar o espaço de trabalho e obter uma resposta à sensação de movimento mais próxima da referência (KHUSRO et al., 2020). No entanto, a aplicação do Controle Preditivo Baseado em Modelo ainda apresenta desafios significativos, como a complexidade do modelo e a necessidade de previsões precisas do movimento futuro, o que pode não ser viável em tempo real (QAZANI et al., 2021a). Além disso, certos modelos baseados em MPC não consideram adequadamente a integral do deslocamento linear, resultando em pistas de movimento inconsistentes, especialmente em cenários complexos como ambientes urbanos. Para contornar essas deficiências, Qazani et al. (2021a) propôs uma modificação no modelo de MPC, incorporando explicitamente a integral do deslocamento linear para calibrar os parâmetros do sistema. Essa abordagem permitiu uma melhora considerável na fidelidade da resposta do sistema, evidenciada por reduções nos valores de erro quadrático médio e maior correlação entre os sinais do simulador e os esperados pelo usuário. Ainda que os resultados sejam encorajadores, o desempenho do MPC permanece altamente dependente da qualidade da previsão futura, o que limita sua aplicação prática imediata.

Para demonstrar a utilização prática do MCA em plataformas de 2 graus de liberdade, o trabalho de Nehaoua et al. (2006) comparou 3 metodologias de algoritmos aplicados a uma plataforma de movimento com 2 graus de liberdade, focando na restituição da aceleração longitudinal. Três abordagens diferentes foram comparadas: a clássica, a adaptativa e a ótima. Os resultados mostraram que o algoritmo clássico é simples de implementar bem como ajustar, possuindo resposta rápida e eficiente. No entanto, ele pode induzir percepções incorretas durante variações bruscas de aceleração, como frenagens intensas. Já o algoritmo adaptativo apresentou uma resposta mais suave, porém com retorno mais lento à posição neutra, o que pode afetar a sensação de realismo em manobras rápidas. Por fim, o algoritmo ótimo, apesar

de minimizar o erro entre o movimento real e o simulado, não conseguiu proporcionar uma percepção realista devido às limitações físicas da plataforma utilizada.

Devido à estrutura simples do filtro *washout*, seu fácil ajuste e baixa demanda computacional, o MCA clássico se torna extremamente adequado a aplicações comerciais que exigem estabilidade e resposta rápida do sistema, como em simulações em tempo real (MURGOVSKI, 2007). No entanto, por manter parâmetros fixos durante toda a operação, pode haver um uso limitado do espaço físico da plataforma. Com o objetivo de contornar essa limitação, Qazani et al. (2021b) propôs uma metodologia adaptativa utilizando uma frequência de corte variável ao longo do tempo. Embora essa adaptação ofereça maior flexibilidade, sua implementação pode se tornar instável ou difícil de calibrar quando baseada apenas em regras fixas ou modelos determinísticos. Para lidar com essa complexidade, o autor incorporou unidades de lógica *fuzzy*, capazes de ajustar os filtros com base em informações como o erro de sensação entre o condutor real e o usuário do simulador, além da posição e condição do efetuador final. Aplicada a uma plataforma Stewart em um ambiente simulado, a proposta inclui ainda um mecanismo de detecção de direção, que contribui para uma atuação mais dinâmica do sistema. Como resultado, o modelo apresentou um uso mais eficiente do espaço de trabalho da plataforma.

Embora modelos adaptativos e técnicas avançadas possam oferecer melhorias em cenários ideais, este trabalho optou pelo MCA clássico devido às limitações físicas da plataforma. Observou-se que o peso da carga durante o movimento gera desbalanceamentos capazes de comprometer a estabilidade do sistema, causando vibrações significativas na base. Como essas restrições já limitam naturalmente o espaço de trabalho, o uso de algoritmos mais complexos traria apenas ganhos pouco significativos em comparação com a solução clássica. Além disso, a escolha se fundamenta na simplicidade estrutural, confiabilidade e facilidade de aplicação em contextos industriais (QAZANI et al., 2021b), nos quais a robustez e a adaptabilidade prática são essenciais em um modelo piloto de simulador de movimento para treinamento de operadores de caminhão fora de estrada. Como alternativa à calibração empírica tradicional, foi empregada uma abordagem determinística baseada em análise paramétrica dos parâmetros dos filtros do modelo MCA, permitindo a seleção de um ponto de operação fixo que atende adequadamente aos requisitos da aplicação. Essa estratégia visa fornecer uma solução tecnicamente viável e facilmente integrável, com foco na criação de ambientes de simulação seguros, consistentes e operacionais para o treinamento de operadores de veículos pesados operados em área de mina.

2.3 Avaliação de desempenho do MCA

A avaliação da qualidade dos algoritmos de sensação de movimento ainda é um desafio na área de simulação, principalmente pela ausência de uma metodologia padronizada que seja amplamente reconhecida e aplicada (CASAS-YRURZUM et al., 2020). Abordagens subjetivas, consistem na realização de testes práticos com operadores e coleta de suas impressões por meio de questionários ou entrevistas ainda são comuns, sobretudo em aplicações comerciais e de treinamento (REYMOND; KEMENY, 2000). Apesar de oferecerem uma percepção direta do conforto e realismo experimentado pelo usuário, essas metodologias apresentam forte dependência do perfil individual dos participantes, variando significativamente conforme a experiência prévia, sensibilidade ao movimento e expectativas pessoais. Além disso, a falta de controle sobre variáveis psicológicas e fisiológicas torna difícil a repetibilidade dos testes e compromete a validade estatística dos resultados obtidos.

Outra abordagem amplamente utilizada na avaliação de algoritmos de sensação de movimento é baseada na modelagem do sistema vestibular humano. O objetivo é estimar como os estímulos físicos gerados pela plataforma são percebidos pelo usuário, considerando as respostas dos órgãos otolíticos e dos canais semicirculares. Essa estratégia fornece uma perspectiva fisiologicamente fundamentada, permitindo que os estímulos simulados sejam comparados com os reais sob a ótica da percepção vestibular.

Embora essa técnica ofereça uma visão mais sistemática do problema e seja fundamentada em bases fisiológicas sólidas, sua aplicação prática ainda é limitada (ASADI et al., 2016). Os modelos vestibulares, por serem simplificações do funcionamento real do sistema sensorial humano, não capturam com precisão toda a complexidade da percepção de movimento.

Um exemplo de modelagem bastante utilizado é proposto por Rengifo et al. (2021), que descreve o comportamento dos canais semicirculares por meio da seguinte Equação 2.1.

$$\frac{\hat{\omega}(s)}{\omega(s)} = \frac{\tau_L \tau_a s^2 (1 + \tau_l s)}{(1 + \tau_a s)(1 + \tau_L s)(1 + \tau_s s)} \quad (2.1)$$

Onde $\omega(s)$ é a velocidade angular percebida, $\hat{\omega}(s)$ é a velocidade angular real, τ são constantes de tempo que podem ser vistos na Tabela 1. Para os otólitos, a relação entre a força específica real f e a força percebida $\hat{f}$ é dada por (Equação 2.2):

$$\frac{\hat{f}(s)}{f(s)} = G_o \frac{\tau_a o s + 1}{(\tau_{Lo} s + 1)(\tau_{so} s + 1)} \quad (2.2)$$

Onde os parâmetros τ e G_o definem as características dinâmicas do sistema vestibular. Estes e demais parâmetros são apresentados na Tabela 1, onde M1 é o

modelo adotado por Young e Oman (1969) e M2 é o modelo adotado por Houck, Telban e Cardullo (2005).

Tabela 1 – Parâmetros do modelo vestibular

Modelo	τ_L	τ_s	τ_l	τ_a	G_o	τ_{ao}	τ_{Lo}	τ_{so}
M1	x: 6,1s							
	y: 5,3s	0,1s	0s	30s	0,4	13,2s	5,33s	0,66s
	z: 10,2s							
M2	5,73s	0,005s	0,06s	80s	0,4	10s	5s	0,016s

Contudo, há pouca padronização na escolha dos modelos e nos critérios utilizados para comparar estímulos reais e simulados, o que dificulta a comparação entre estudos e limita seu uso como ferramenta de validação robusta. Apesar disso, os modelos vestibulares ainda se mostram eficazes como uma forma direta para avaliar a fidelidade de movimentos em algoritmos MCA, sendo esta uma das abordagens adotadas no presente trabalho utilizando o modelo proposto por Houck, Telban e Cardullo (2005), que será apresentado com mais detalhes na Seção 3.

Outra análise objetiva do desempenho da plataforma e de fácil implementação é o *Objective Motion Cueing Test* (OMCT). Esse teste consiste na aplicação de sinais controlados em cada grau de liberdade da base de movimento, possibilitando a obtenção da resposta em frequência do sistema (SEEHOF; DURAK; DUDA, 2014). A partir dessa análise, torna-se viável identificar perdas de ganho, atrasos, distorções e acoplamentos cruzados entre eixos, elementos críticos para a qualidade da simulação. Por gerar métricas quantitativas, reproduzíveis e independentes da percepção subjetiva do operador, o OMCT permite comparações diretas entre diferentes configurações de simuladores e algoritmos. Justamente por reunir essas características, aliadas à sua simplicidade de aplicação, o OMCT tornou-se o método mais utilizado na literatura e, por essa razão, é o adotado neste trabalho para avaliação do sistema estudado.

O OMCT consiste na aplicação de sinais senoidais de diferentes frequências na entrada do sistema de movimento, com o objetivo de analisar a resposta gerada na saída (HOSMAN; ADVANI, 2016). As frequências são criteriosamente selecionadas para cobrir o espectro de interesse, permitindo uma caracterização mais precisa do comportamento dinâmico da plataforma. Ao aplicar cada frequência individualmente, é possível observar com clareza os efeitos isolados de cada componente do sinal sobre o sistema.

3 Sistema vestibular

O sistema vestibular é um órgão sensorial localizado no interior do sistema auditivo, abrigado na região petrosa do osso temporal, ilustrado na Figura 7. Ele desempenha um papel fundamental na manutenção do equilíbrio corporal, na percepção de orientação espacial e na estabilização do olhar durante o movimento. Anatomicamente, essa estrutura é composta por duas partes principais: o labirinto ósseo, constituído por túneis e cavidades esculpidos no osso temporal, e o labirinto membranoso, que é uma estrutura mais interna e flexível, suspensa no interior do labirinto ósseo (BEZERRIL; OLIVEIRA, 2022). O labirinto ósseo contém três canais semicirculares (anterior, posterior e lateral), o vestíbulo e a cóclea. Os canais semicirculares e o vestíbulo formam a porção funcional do aparelho vestibular, sendo responsáveis pela detecção de acelerações angulares e lineares, respectivamente, parte responsável pelas sensações físicas de movimento. Já a cóclea, embora compartilhe a localização anatômica, é especializada na audição e tem pouca relação funcional com o equilíbrio (BEZERRIL; OLIVEIRA, 2022).

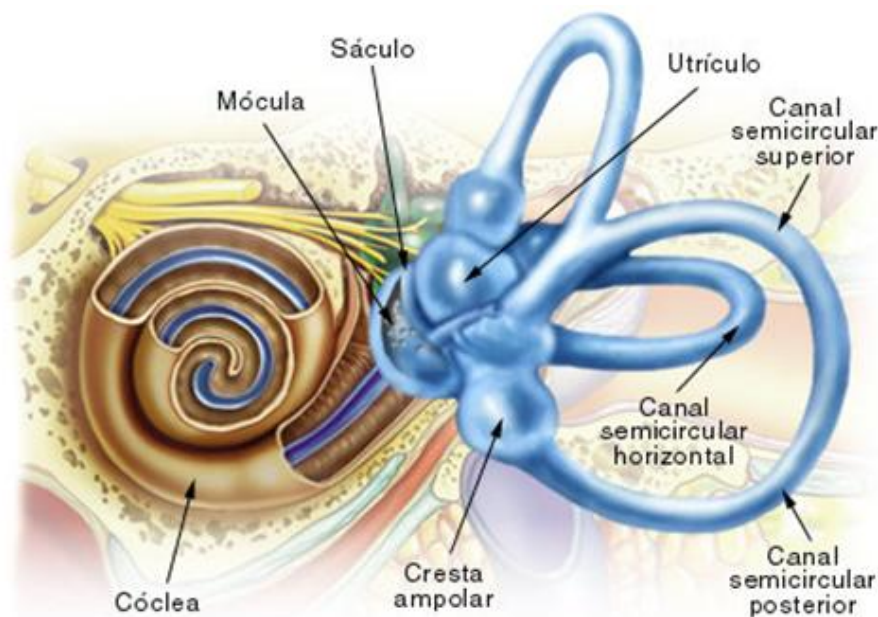


Figura 7 – Sistema vestibular humano (PEREIRA, 2021).

3.1 Canais semicirculares

Os três canais semicirculares são estruturas sensoriais orientadas aproximadamente em planos ortogonais entre si, permitindo a detecção de rotações angulares da cabeça

em todas as direções do espaço.

Cada canal contém um fluido chamado endolinfa e possui uma região dilatada chamada ampola, onde ficam as células sensoriais que detectam o movimento. Quando a cabeça gira, o canal se move junto com o crânio, mas o fluido dentro dele tende a ficar parado por inércia. Essa diferença entre o movimento do canal e do fluido provoca a curvatura de uma estrutura chamada cúpula, localizada na ampola, mostrado na Figura 8. Essa curvatura é percebida pelas células sensoriais, que enviam sinais para o cérebro indicando que houve uma rotação, em que direção e com que velocidade.

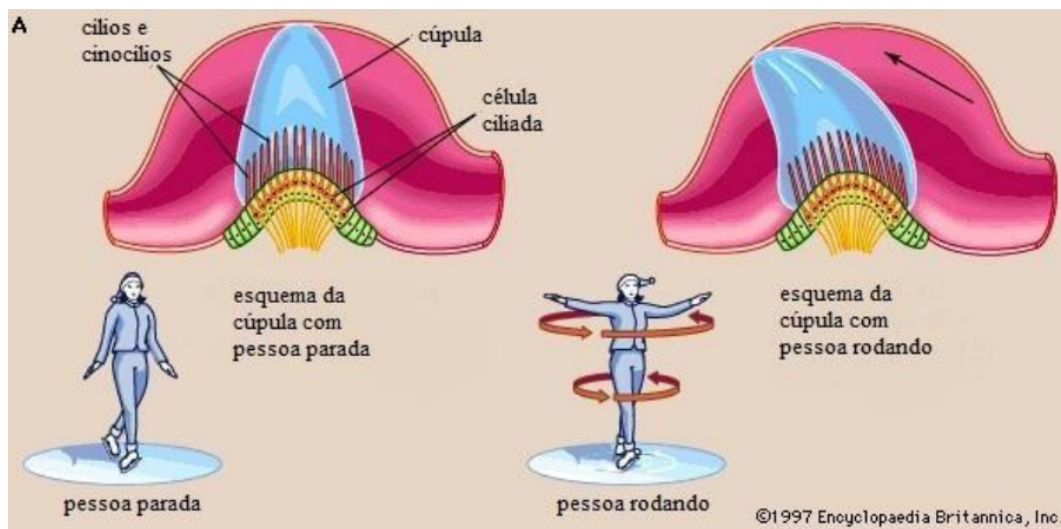


Figura 8 – Cílio do canal semicircular (BAZONI, 2016).

A contribuição de cada canal depende do plano do movimento. O canal lateral é ativado principalmente quando giramos a cabeça para dizer “não”, ou seja, no plano horizontal. O canal anterior responde a movimentos como quando assentimos com a cabeça, como ao dizer “sim”. Já o canal posterior está mais relacionado a inclinações para o ombro, como quando se encosta a orelha no ombro (ANGELAKI; CULLEN, 2008). Juntos, os três canais permitem que o cérebro tenha uma noção precisa da orientação da cabeça durante movimentos rotacionais. Essa informação é essencial para manter o equilíbrio e para que os olhos possam se mover de forma compensatória, estabilizando a visão mesmo quando o corpo está em movimento.

Segundo (HOUCK; TELBAN; CARDULLO, 2005), o canal semicircular pode ser simplificado por uma função de transferência de segunda ordem dada por (Equação 3.1):

$$\frac{\hat{\omega}_i(s)}{\omega_{i,rot}(s)} = \frac{80s^2}{(1 + 80s)(1 + 5,73s)} \quad (3.1)$$

3.2 Órgãos otolíticos

Os órgãos otolíticos, que são o utrículo e o sáculo, são estruturas especializadas na detecção de acelerações lineares e na orientação da cabeça em relação à gravidade. Cada uma delas contém uma mácula, região sensorial composta por células ciliadas recobertas por uma membrana gelatinosa, sobre a qual repousam cristais de carbonato de cálcio denominados otólitos (BEZERRIL; OLIVEIRA, 2022). Quando o corpo sofre uma aceleração linear ou muda de inclinação, a inércia dos otólitos causa o deslocamento da membrana, defletindo os cílios das células sensoriais como na Figura 9. O utrículo é mais sensível a movimentos horizontais (como acelerar em linha reta), enquanto o sáculo responde melhor a movimentos verticais (como subir ou descer em um elevador) (ANGELAKI; CULLEN, 2008). Esses sinais são essenciais para a manutenção do equilíbrio estático e para o controle postural em situações dinâmicas.

Segundo Houck, Telban e Cardullo (2005), a função de transferência que representa este órgão é dada pela Equação 2.2:

$$\frac{\hat{f}_i(s)}{f_i(s)} = 0,4 \frac{1 + 10s}{(1 + 5s)(1 + 0,016s)}$$

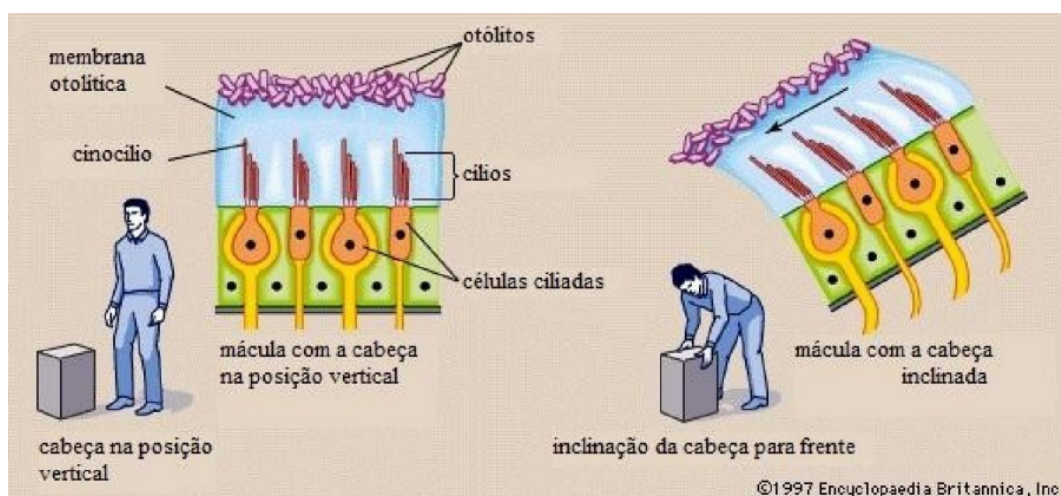


Figura 9 – Movimento do sistema otólito (BAZONI, 2016).

3.3 Limiar de sensação de movimento

A definição dos valores limiares de percepção (*threshold*) é fundamental para compreender os limites de detecção do sistema vestibular humano diante de estímulos rotacionais e lineares. Esses limiares representam os menores níveis de aceleração que podem ser percebidos conscientemente pelos canais semicirculares e órgãos otólitos, sendo especialmente relevantes na modelagem dos acoplamentos cruzados do MCA,

uma vez que é o movimento de aceleração translacional que precisa ser representado por uma inclinação, sem que os canais semicirculares detectem o movimento de rotação.

Houck, Telban e Cardullo (2005) reuniram, a partir de diferentes fontes experimentais, uma série de valores representativos desses valores de *threshold*. A Tabela 2 apresenta os valores mais comumente utilizados na literatura, encontrados pelos autores, para os limiares de percepção associados a movimentos rotacionais e translacionais.

Tabela 2 – Valores típicos de threshold de percepção para movimentos

Grau de Liberdade	Valor de Limiar
Velocidade Angular em <i>Roll</i>	3,0°/s
Velocidade Angular em <i>Pitch</i>	3,6°/s
Velocidade Angular em <i>Yaw</i>	2,6°/s
Aceleração Linear Longitudinal	0,17 m/s ²
Aceleração Linear Lateral	0,17 m/s ²
Aceleração Linear Vertical	0,28 m/s ²

4 Plataforma de movimentos

Bases de movimento são componentes fundamentais em simuladores de direção, pois permitem a reprodução física de inclinações e acelerações percebidas pelo usuário. Embora plataformas com múltiplos graus de liberdade, como a de Stewart, sejam amplamente empregadas em diversas aplicações, configurações mais simples podem atender de forma eficaz a contextos específicos. No caso de caminhões fora de estrada, veículos com dinâmica mais lenta e movimentos predominantes de rolagem e inclinação, bases com dois graus de liberdade (2DoF) oferecem uma solução adequada para fins de treinamento, combinando realismo com simplicidade construtiva.

A escolha por uma base com 2DoF permite a reprodução de movimentos nos eixos de *pitch* e *roll*, que estão entre os mais relevantes para simular as sensações experimentadas na condução de veículos terrestres. Devido à sua arquitetura simplificada, esse tipo de base possibilita a construção de sistemas mais econômicos e com manutenção facilitada, sem prejudicar a eficácia do treinamento. O estudo de Arioui, Nehaoua e Amouri (2005) mostra que resultados estáveis podem ser obtidos com algoritmos de controle de movimento aplicados a plataformas de baixo custo com dois graus de liberdade.

O trabalho apresentado por Zhang e Zhang (2013) implementa uma plataforma de movimento muito similar à utilizada nesta dissertação, que pode ser visto na Figura 10. Em sua abordagem, o autor realiza o cálculo da cinemática de uma base paralela com dois graus de liberdade, composta por três pontos fixos na base (denotados por A_1 , A_2 , A_3) e três pontos móveis na plataforma superior (B_1 , B_2 , B_3), como pode ser visto na Figura 11. A conexão entre esses pontos é feita por atuadores elétricos que transladam verticalmente, permitindo que a plataforma superior execute rotações em torno dos eixos X e Y . A geometria do sistema é tal que o ponto A_1B_1 funciona como eixo de rotação fixo, enquanto os demais pontos (A_2B_2 e A_3B_3) são controlados por atuadores.

4.1 Cinemática da plataforma de 2DoF

A cinemática direta de manipuladores paralelos é, em geral, de difícil resolução analítica devido à complexidade geométrica e às múltiplas restrições envolvidas. A solução para esse problema, além de não ser única, em muitos casos, o tempo de cálculo dessas soluções é elevado demais para aplicações em tempo real (MERLET, 2005). Por outro lado, a cinemática inversa desses mecanismos tende a ser mais acessível e bem definida. Ela busca determinar o quanto cada atuador deve se alongar

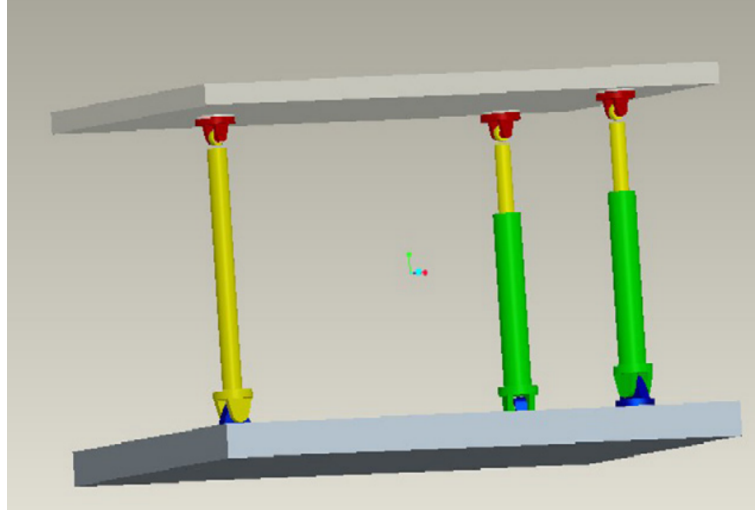


Figura 10 – Diagrama 3D da base de movimento utilizada por Zhang e Zhang (2013).

ou retrain para atingir uma determinada orientação da plataforma. O cálculo parte da posição inicial dos pontos móveis e aplica uma transformação para obter suas novas posições após a rotação desejada, conforme realizado por Zhang e Zhang (2013) em seu projeto de uma plataforma de 2 graus de liberdade, similar à utilizada nesta dissertação. A diferença entre as posições transformadas e os pontos de ancoragem na base define o deslocamento necessário de cada atuador. Desta forma, para calcular a cinemática inversa é necessário calcular os pontos de apoio da plataforma móvel B, apresentada na Figura 11, após o movimento de rotação.

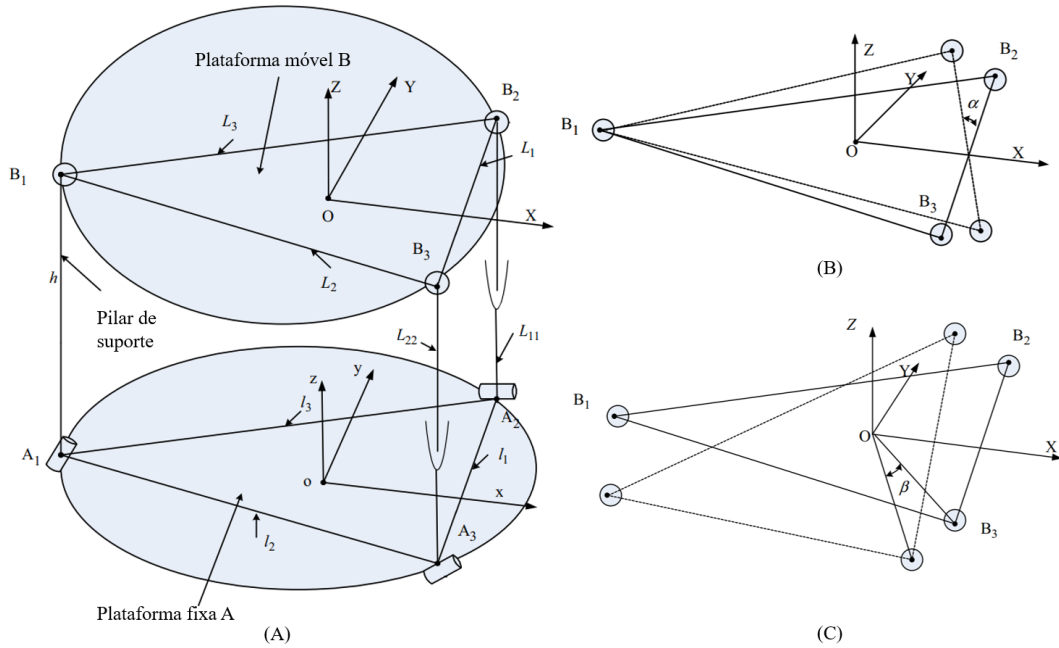


Figura 11 – (A) Diagrama esquemático da plataforma de 2DoF. (B) movimento de *roll* (C) movimento de *pitch*. Adaptado de Zhang e Zhang (2013)

A posição transformada de um ponto B_i da plataforma é dada por:

$$\vec{P}_i = R \cdot \vec{B}_i \quad (4.1)$$

Onde $\vec{B}_i$ é a posição original do ponto na plataforma e R é a matriz de transformação de rotação correspondente às rotações desejadas e P_i é equivalente aos pontos $\vec{B}_i$ após a rotação: Figura 11.

A matriz de transformação R representa a composição de rotações em torno dos eixos X (ângulo α) e Y (ângulo β). Assumindo pequenas rotações e que a plataforma não realiza translações, a matriz homogênea R pode ser escrita como:

$$R = \begin{bmatrix} \cos \beta & \sin \beta \sin \alpha & \sin \beta \cos \alpha & 0 \\ 0 & \cos \alpha & -\sin \alpha & 0 \\ -\sin \beta & \cos \beta \sin \alpha & \cos \beta \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.2)$$

Com essa matriz, os pontos B_i são transformados para obter a nova posição dos pontos B_i dada por $\vec{P}_i$, e então calcula-se a variação de comprimento de cada atuador ΔL_i .

$$\Delta L_i = \|\vec{P}_i - \vec{A}_i\| \quad (4.3)$$

Sendo $\vec{A}_i$ a posição do ponto correspondente na base fixa A .

4.2 Projeto da plataforma de movimentos

Este trabalho apresenta o desenvolvimento de uma base de movimento 2DoF projetada para simular a operação de caminhões fora de estrada, com capacidade para suportar cargas úteis de até 500 kg. Sua estrutura inclui um eixo cardã central fixo, cuja função principal é fornecer estabilidade e suportar o peso da carga, ao mesmo tempo que permite liberdade de movimento. Complementando essa estrutura, dois eixos móveis posicionados na parte traseira são acionados por servomotores, viabilizando os movimentos da base superior nos eixos de *roll* e *pitch*, essenciais para a reprodução da dinâmica veicular (Figura 12).

Com essa configuração, a plataforma pode alcançar uma amplitude angular de até 25°. Contudo, considerando a segurança dos operadores e a preservação dos componentes mecânicos, o limite operacional foi restringido a 20°.

O simulador de operação de mina integra um conjunto robusto de *hardware* e *software*, projetado para reproduzir fielmente as condições reais de trabalho. No nível computacional, a coordenação da plataforma é realizada por um sistema baseado no ROS 2 (MACENSKI et al., 2022), um conjunto de bibliotecas e ferramentas de código aberto voltado ao desenvolvimento de aplicações robóticas. No contexto deste

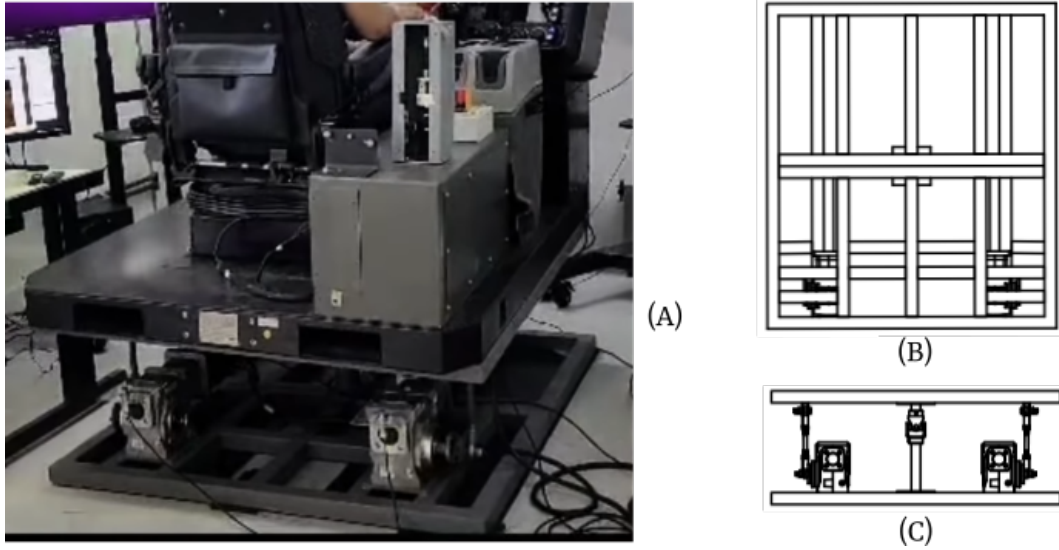


Figura 12 – Plataforma de movimentos com 2DoF desenvolvida. (A) Estrutura real com atuadores e sensores, (B) Vista superior e (C) Vista posterior.

projeto, o ROS 2 foi implementado como base para a comunicação entre os módulos, a gestão dos tópicos de sensores e atuadores, a execução do algoritmo de sensação de movimento e o gerenciamento dos sinais do OMCT, além da coleta e registro de dados para análises posteriores. A Figura 13 representa um diagrama do processo do ROS. A comunicação entre os diversos subsistemas ocorre por meio de uma rede Ethernet, conectando um computador central de alto desempenho a um sistema embarcado *Odroid*, que atua no controle em tempo real da base de movimento. Essa arquitetura distribuída garante sincronização, baixa latência e robustez mesmo em condições de operação com alta demanda dinâmica.

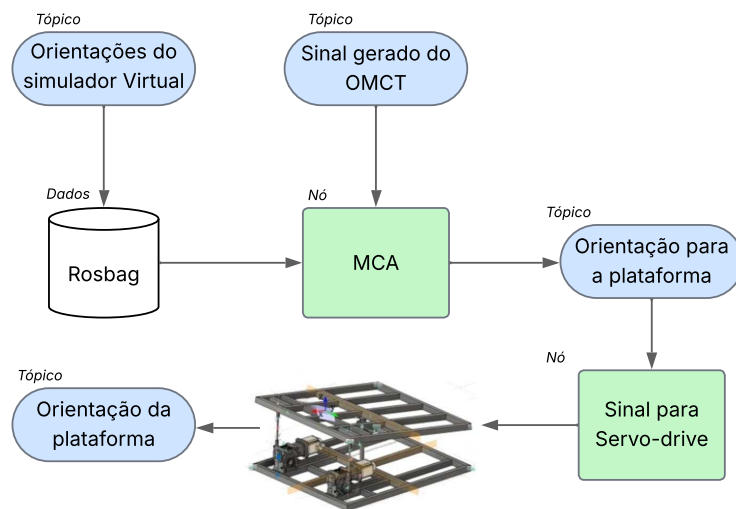


Figura 13 – Diagrama de nós e tópicos do ROS

O *Odroid* desempenha um papel fundamental como servidor de intermediação,

processando os comandos enviados pela unidade principal e repassando as informações para a base de movimento. Ele também recebe os sinais da cabine de operação e os sincroniza com o ambiente virtual da simulação. Para o controle preciso dos motores, os sinais de comando convertidos pelo *Odroid* são transmitidos ao servodrive através de uma interface USB-CAN, utilizando o protocolo CANopen. Esse protocolo garante um envio confiável de comandos e feedback dos atuadores, permitindo ajustes dinâmicos conforme a necessidade da simulação. A Figura 14 ilustra o diagrama de conexões desse subsistema.

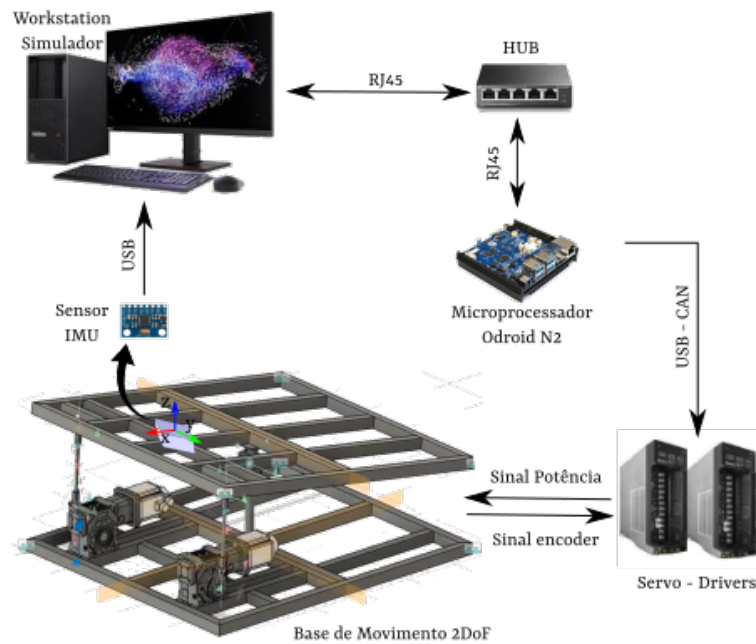


Figura 14 – Plataforma de Hardware e comunicação entre os módulos.

Durante a operação, os servomotores realizam os movimentos programados dos servo-drivers, que ajustam a potência fornecida com base no feedback recebido dos encoders acoplados a cada eixo. Para fornecer uma referência confiável de posição, dois sensores indutivos foram instalados na parte traseira da plataforma, sobre os motores, servindo como um ponto de calibração inicial e delimitando uma região segura de operação. Um acelerômetro foi instalado na base superior da plataforma, próximo ao assento do operador, com a função de monitorar continuamente os ângulos de *roll* e *pitch*. As medições obtidas são transmitidas ao simulador por meio de comunicação serial e registradas para análise e validação do desempenho dinâmico da plataforma. Como pode ser observado, os eixos de referência do acelerômetro foram posicionadas de maneira que o plano XY coincidissem com o plano da base de movimento, tendo a orientação do eixo X apontando para o sentido oposto da frente da base de movimento, paralelo ao eixo longitudinal da plataforma, e o eixo Y sendo paralelo ao seu eixo transversal.

5 Algoritmo de sensação de movimentos (MCA)

5.1 Definição do MCA

O MCA é o componente responsável por transformar os dados provenientes de simulações virtuais em comandos físicos que movimentam a plataforma de forma a replicar sensações de movimento realistas para o usuário (ARANGO et al., 2025). Essa conversão precisa considerar as limitações mecânicas da plataforma e a percepção humana, utilizando estratégias que preservem a fidelidade sensorial sem comprometer a segurança ou causar desconforto. Assim, o desenvolvimento de um MCA eficaz envolve o equilíbrio entre desempenho computacional, resposta dinâmica da plataforma e coerência perceptiva dos movimentos simulados.

Para lidar com estes desafios, diversas abordagens podem ser utilizadas para realizar essa conversão dos movimentos simulados para comandos da plataforma, variando de estratégias baseadas em filtros lineares até métodos mais complexos que incorporam modelos preditivos (ZHAO et al., 2025). No entanto, o método mais amplamente empregado é o filtro clássico de *washout* (*Classical Washout Filter* – CWF), devido à sua simplicidade e eficiência computacional (STUPANS et al., 2024; QAZANI et al., 2021b).

O CWF opera por meio de uma estrutura de filtragem em múltiplos canais, como pode ser visto na Figura 15. Neste diagrama, f_a representa a aceleração linear do veículo, enquanto ω corresponde à velocidade angular. As matrizes R e R_ω são responsáveis por transformar os sinais filtrados para o sistema de coordenadas da plataforma, sendo R aplicada aos movimentos lineares e R_ω aos movimentos angulares. Como resultado do processamento, obtêm-se os sinais t_d e θ_d , que representam, respectivamente, os comandos de translação e rotação a serem enviados à base de movimento. No caso do sistema considerado neste trabalho, apenas os componentes rotacionais de *roll* e *pitch*, contidos em θ_d , serão utilizados para acionar a plataforma.

Diferentes tipos de movimento são processados separadamente por cadeias de filtros divididos em três canais:

O canal translacional: é responsável por processar as acelerações lineares nos eixos X, Y e Z em um simulador de movimento. Seu principal objetivo é atenuar as componentes de baixa frequência dessas acelerações, para evitar deslocamentos muito grandes da base móvel, ultrapassando seus limites físicos de atuação (LEMES, 2018). Como a plataforma utilizada neste trabalho possui apenas dois graus de liberdade rotacionais (*roll* e *pitch*), este canal não foi implementado. A estrutura do filtro

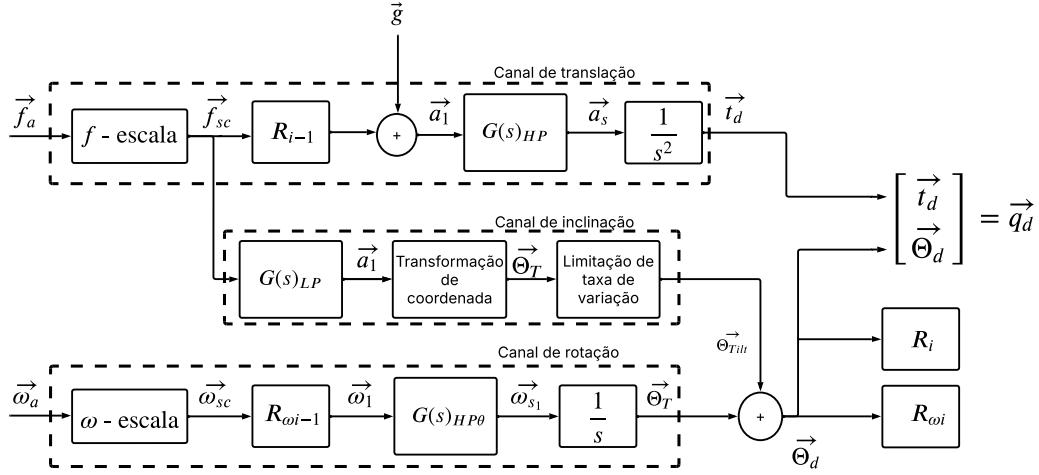


Figura 15 – MCA clássico, adaptado de Vargas (2009).

passa-alta é apresentada na Equação (5.1):

$$G(s)_{HP} = \frac{s^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (5.1)$$

Canal de inclinação: O canal de inclinação tem como objetivo simular acelerações lineares de baixa frequência por meio da inclinação controlada da plataforma, técnica conhecida como *tilt coordination*. Esse canal atua sobre os sinais de aceleração nos eixos X e Y, aplicando filtros passa-baixa para remover componentes de alta frequência, mantendo apenas as variações lentas que podem ser reproduzidas de forma imperceptível através da inclinação do sistema. Ao inclinar a base, a componente da força gravitacional passa a representar uma aceleração linear contínua, sem exigir grandes deslocamentos físicos da plataforma. Para que o operador não perceba essa inclinação, impõe-se um limite de variação angular, usualmente de até 3 graus por segundo (SPRINGER; GATTRINGER; BREMER, 2011). A estrutura do filtro passa-baixa utilizado está apresentada na Equação (5.2):

$$G(s)_{LP} = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (5.2)$$

Para aplicar a inclinação proveniente do acoplamento cruzado, é necessário determinar a componente da força gravitacional gerada pela inclinação da cadeira, de modo que ela represente a mesma sensação da força de aceleração do caminhão. A relação de forças pode ser visto na Figura 16 e a inclinação pode ser obtida igualando as Equações (5.3) e (5.4).

$$F = ma \quad (5.3)$$

$$F_g = mg \sin(\theta_{tilt}) \quad (5.4)$$

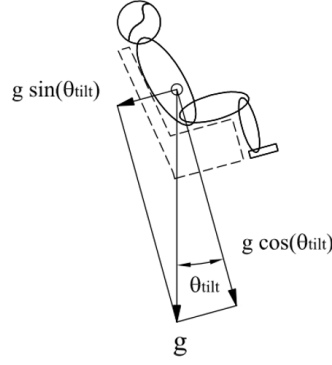


Figura 16 – Avaliação do valor de ω_n em $G(s)_{HP0}$, adaptado de Vargas (2009)).

Igualando as forças e isolando o ângulo θ tem-se a Equação (5.5):

$$\text{sen}(\theta_{\text{tilt}}) = \frac{a}{g} \quad (5.5)$$

Para ângulos pequenos $\text{sen}(\theta) = \theta$, a Equação (5.6) apresenta:

$$\theta_{\text{tilt}} \approx \frac{a}{g} \quad (5.6)$$

O canal rotacional: O canal de rotação é responsável por processar os sinais de velocidade angular ω , correspondentes aos movimentos de *roll* e *pitch*, com o objetivo de simular apenas as variações rotacionais de alta frequência. Para isso, são aplicados filtros passa-alta que atenuam os componentes lentos e preservam apenas as mudanças rápidas de orientação, que são então integradas para gerar os deslocamentos angulares. A estrutura do filtro passa-alta utilizado nesse canal é apresentada na Equação (5.7):

$$G(s)_{HP\theta} = \frac{s}{s + \omega_n} \quad (5.7)$$

5.2 Parametrização do MCA

Para o projeto do MCA da plataforma de 2 graus de liberdade é necessário definir os parâmetros ζ e ω_n do filtro $G(s)_{LP}$ e ω_n do filtro $G(s)_{HP\theta}$. Durante o funcionamento da plataforma, verificou-se que ela apresenta maior estabilidade em torno de 0,5 Hz (3,5 rad/s). Por esse motivo, essa faixa de frequência foi selecionada como referência para o dimensionamento dos filtros. Para definir o valor de ζ foi mantido o valor de ω_n unitário e traçadas várias curvas com variação de ζ de 0,5 a 1,5, o resultado pode ser visto na Figura 17.

Em um sistema ideal, o filtro passa-baixa aplicado no canal de inclinação deve apresentar ganho unitário nas frequências mais baixas, garantindo que os movimentos lentos do sistema de entrada sejam replicados de forma fiel na saída. À medida que

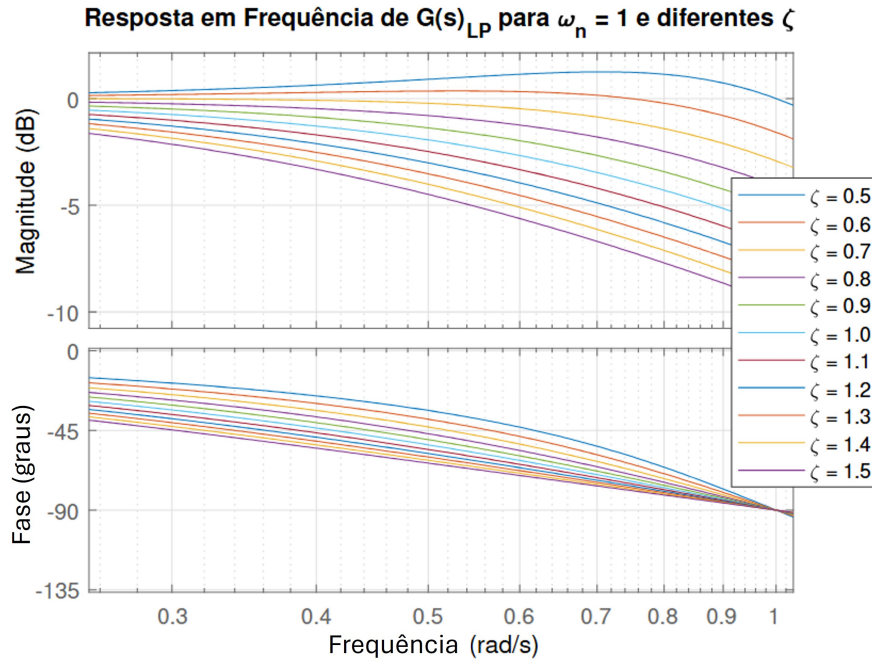


Figura 17 – Avaliação do valor de ζ em $G(s)_{LP}$

a frequência aumenta, o filtro deve começar a atenuar progressivamente os sinais, limitando a resposta do sistema apenas às componentes desejadas. Todas as curvas apresentadas na Figura 17 seguem o mesmo padrão de comportamento dinâmico. O fator de amortecimento adotado foi $\zeta = 0,7$, uma vez que, nesse valor, a resposta do sistema apresenta uma magnitude próxima de zero em baixas frequências, o que indica uma boa representação do sinal. Para valores de $0,7$, observa-se um aumento nos ganhos em determinadas faixas de frequência, o que pode resultar em amplificações indesejadas e, conseqüentemente, em distorções na resposta do sistema.

Posteriormente, com o ζ definido em $0,7$ foram traçadas várias curvas de variação de ω_n oscilando de 1 até $6,5$, o resultado pode ser visto na Figura 18. Para a função de inclinação no eixo de *roll*, o valor de ω_n adotado foi de 3 rad/s, pois a partir de aproximadamente $3,5$ rad/s (cerca de $0,56$ Hz) o sistema começa a apresentar instabilidade, sendo desejável atenuar os sinais consideravelmente a partir desse ponto, sem comprometer as componentes de baixa frequência. Já para o movimento de *pitch*, que apresenta oscilações mais significativas, foi utilizado um ω_n de $3,5$ rad/s, permitindo acompanhar melhor a dinâmica do sinal sem amplificar ruídos ou gerar desconforto ao operador.

Para o canal de rotação foram traçadas várias curvas de variação de ω_n oscilando de 1 até $6,5$, o resultado pode ser visto na Figura 19. Para o canal de rotação, é desejável preservar as componentes de alta frequência, uma vez que essas variações são mais perceptíveis e contribuem diretamente para a sensação de movimento angular.

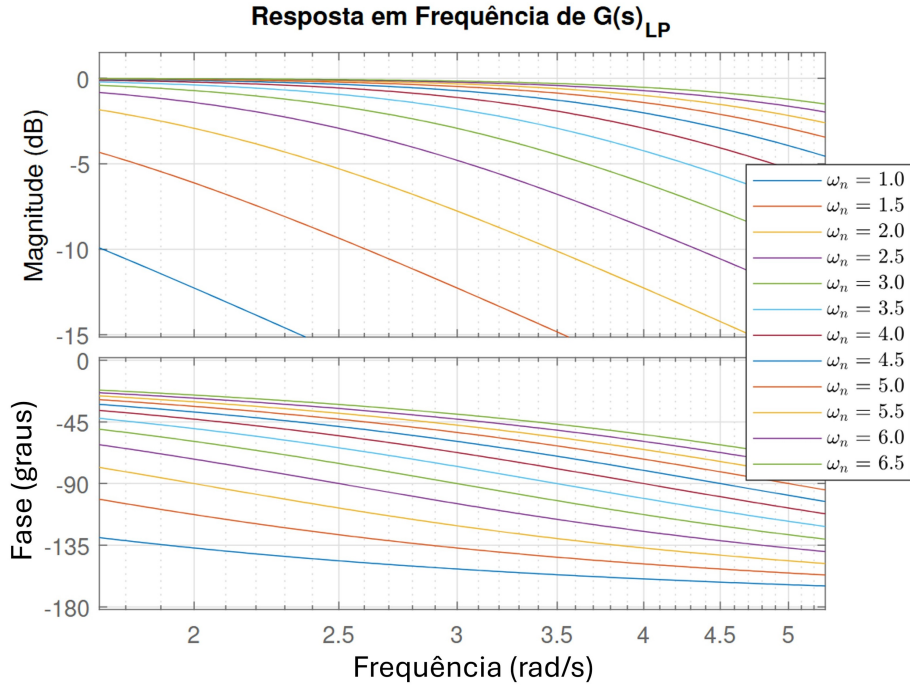


Figura 18 – Avaliação do valor de ω_n em $G(s)_{LP}$

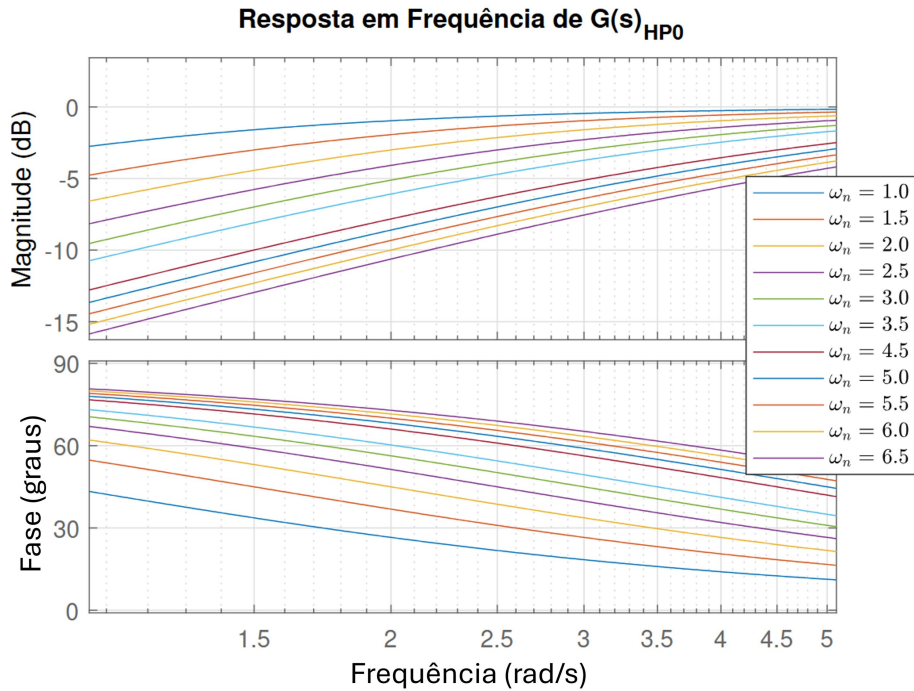


Figura 19 – Avaliação do valor de ω_n em $G(s)_{HP0}$

Dessa forma, o valor de ω_n foi mantido em 1 rad/s, pois atenua adequadamente as frequências abaixo de aproximadamente 0,3 rad/s (cerca de 0,5 Hz), permitindo que as frequências mais altas sejam mantidas com fidelidade.

O MCA desenvolvido para a plataforma de movimentos deste estudo é similar ao proposto por (VARGAS, 2009), originalmente aplicado a uma base com 6 graus de

liberdade. Apesar da diferença no número de eixos, o modelo conceitual é idêntico e pode ser reduzido a plataformas de menor grau de liberdade, como a base de 2DoF adotada aqui. Assim, o algoritmo de sensação de movimento apresentado por (VARGAS, 2009), cujo sistema é idêntico ao utilizado neste trabalho, teve seus parâmetros implementados na plataforma projetada neste projeto para fins de comparação. Os parâmetros de ambos os modelos estão listados na Tabela 3, permitindo uma análise de desempenho frente a uma abordagem já consolidada.

Tabela 3 – Tabela de parâmetros do MCA

Parâmetros	(VARGAS, 2009)	MCA proposto
ζ inclinação X	1	0,7
ω_n inclinação X	5	3
ζ inclinação Y	1	0,7
ω_n inclinação Y	8	3,5
ω_n rotação X	1	1
ω_n rotação Y	1	1

6 Avaliação de desempenho do algoritmo de sensação de movimento

6.1 *Objective Motion Cueing Test* (OMCT)

Como meio de avaliação do algoritmo do MCA, foi escolhido o *Objective Motion Cueing Test* (OMCT), por ser amplamente adotado na literatura e por fornecer uma forma sistemática de analisar o desempenho dinâmico do algoritmo por meio da resposta em frequência (CASAS-YRURZUM et al., 2020).

O OMCT consiste na aplicação de sinais senoidais de diferentes frequências na entrada do sistema de movimento, com o objetivo de analisar a resposta gerada na saída. Conforme as diretrizes recomendadas por Seehof, Durak e Duda (2014), são utilizadas pelo menos doze frequências específicas, distribuídas ao longo do espectro de interesse, para garantir uma avaliação abrangente do comportamento dinâmico do sistema. Durante os testes, cada frequência é aplicada de forma isolada a cada entrada do sistema, permitindo que os efeitos individuais sobre a resposta da plataforma sejam claramente observados.

As respostas aos estímulos são medidas por meio de um acelerômetro fixado diretamente na estrutura móvel da plataforma, de modo a captar com precisão os sinais correspondentes aos movimentos de *roll* e *pitch*. A partir dos dados adquiridos, é realizada uma análise no domínio da frequência, extraindo, para cada valor de excitação, a magnitude e a fase da resposta do sistema. Esses resultados são então organizados na forma de gráficos de resposta em frequência, os quais permitem uma avaliação clara de fenômenos como atrasos, perdas de ganho, distorções e acoplamentos indesejados entre eixos.

A interpretação dos gráficos de resposta em frequência obtidos a partir da simulação permite avaliar a fidelidade com que a plataforma reproduz os sinais simulados. Um ganho próximo de 1 indica que a amplitude do movimento real está coerente com a amplitude do sinal de referência, representando uma boa correspondência entre os sinais reais e simulados. Por outro lado, atrasos de fase observados nos gráficos podem ser interpretados como atrasos temporais na representação do movimento, o que pode comprometer o realismo da simulação, especialmente em aplicações que exigem alta sincronização entre estímulo visual e movimento físico.

Considerando que a plataforma projetada apresenta dois graus de liberdade rotacionais, a aplicação do OMCT requer apenas quatro configurações de teste para cobrir as possíveis interações do sistema. Os dois primeiros testes consistem na aplicação direta de sinais nos eixos *roll* e *pitch*, de forma isolada, com o intuito de

analisar a resposta específica de cada eixo à excitação correspondente. Os outros dois testes analisam o fenômeno de acoplamento cruzado, que é quando os movimentos translacionais aplicados nos eixos X e Y (como acelerações lineares) geram, como resposta, movimentos rotacionais nos eixos correspondentes (*roll* e *pitch*), simulando a percepção inclinada do corpo mesmo sem rotação real. O fluxograma que descreve a sequência de procedimentos utilizados no OMCT está ilustrado na Figura 20.

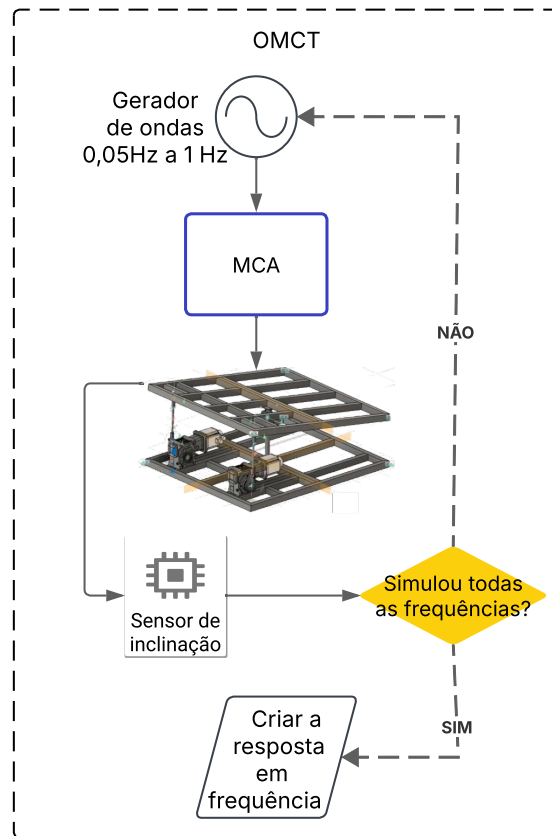


Figura 20 – Fluxograma do teste OMCT

Durante a realização dos testes experimentais, observou-se que, embora a amplitude dos sinais de entrada pudesse ser estendida até aproximadamente 20 graus, esse aumento estava diretamente relacionado a um acréscimo significativo no nível de ruído presente nas respostas do sistema. Tal comportamento é particularmente relevante, uma vez que o aumento do ruído compromete não apenas a precisão dos dados obtidos, mas também pode colocar em risco a integridade estrutural da plataforma, devido à ocorrência de movimentos mais intensos e abruptos, especialmente nas faixas de frequência mais elevadas. Com o objetivo de preservar a estabilidade da estrutura e assegurar a realização segura dos ensaios, optou-se por limitar a amplitude das excitações aplicadas durante o OMCT a uma magnitude máxima de 8 graus, valor considerado adequado para manter a fidelidade da resposta do sistema

sem comprometer sua integridade física.

6.2 Análise da sensação de movimento pelo modelo do sistema vestibular humano

Com o objetivo de avaliar a atuação do MCA desenvolvido para uma plataforma de movimento com dois graus de liberdade, foi adotada uma abordagem baseada na simulação do sistema vestibular humano. A estratégia consiste em estimar a sensação de movimento percebida pelo usuário tanto em um cenário ideal (sem MCA, sem limitações físicas) quanto no cenário real (com MCA e restrições da plataforma), utilizando os modelos fisiológicos que representam os canais otolíticos e semicirculares apresentados por Houck, Telban e Cardullo (2005).

Para a execução do teste, foram utilizadas as acelerações lineares, aplicadas aos modelos dos canais otolíticos, e as velocidades angulares, aplicadas ao modelo dos canais semicirculares, ambas registradas durante a simulação do circuito. Esses sinais foram processados pelas funções de transferência dos respectivos modelos vestibulares. O mesmo conjunto de entradas foi utilizado em dois contextos distintos: um diretamente, representando o movimento original do veículo, e outro após o processamento pelo algoritmo de MCA, refletindo o movimento efetivamente reproduzido pela plataforma. Esse fluxo de processamento é ilustrado na Figura 21, onde e representa o erro quadrático médio calculado entre os dois sinais simulados.

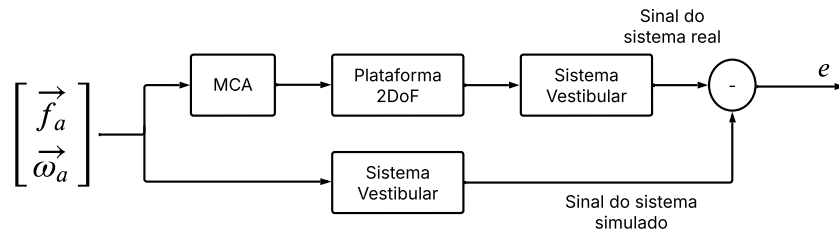


Figura 21 – Fluxograma análise de desempenho do MCA usando modelo Vestibular

Durante esse processo, observou-se que os sinais obtidos a partir do sensor da plataforma apresentaram pequenas inconsistências temporais devido a limitações técnicas, como atrasos de comunicação ou ruídos momentâneos. Para lidar com essas distorções e permitir uma comparação mais precisa entre os sinais, foi aplicado o algoritmo *Dynamic Time Warping* (DTW), que é um algoritmo baseado em programação dinâmica utilizado para medir a similaridade entre duas sequências temporais que podem apresentar variações de tempo ou velocidade. Ele calcula a menor distância possível entre os pontos das duas séries, mesmo quando estão desalinhadas no tempo (YADAV; ALAM, 2018).

Após o alinhamento dos sinais com o DTW, foi possível minimizar os efeitos de defasagens temporais e distorções nos sinais adquiridos, permitindo uma comparação mais fiel entre os cenários com e sem MCA. A partir desse alinhamento, além do cálculo do erro quadrático médio (RMSE), foi também determinado o coeficiente de correlação (CC) entre os sinais processados pelos modelos vestibulares, fornecendo uma avaliação complementar da diferença e da semelhança entre os cenários analisados. Esses resultados tornam possível comparar, de forma objetiva, a sensação de movimento esperada na simulação com aquela efetivamente gerada pela plataforma.

6.3 Caracterização do Ruído da plataforma de movimentos

Durante a operação de plataformas de movimento, a presença de ruídos nos sinais capturados é um fator que pode impactar diretamente a qualidade da simulação e a fidelidade da resposta percebida. Os ruídos podem surgir tanto de limitações dos sensores quanto de efeitos mecânicos da estrutura, influenciando diretamente a estabilidade do sistema e a qualidade da simulação sensorial (RANAWEERA; MAHMOUD, 2022).

Entre as causas de ruído, destacam-se os movimentos de alta frequência, nos quais a estrutura ainda não se estabilizou completamente antes da próxima variação de movimento. Isso resulta em uma superposição de acelerações ou orientações, refletindo em oscilações no sinal medido. Em situações desse tipo, o tempo de resposta da plataforma e a dinâmica do sistema interferem diretamente na precisão do movimento simulado.

A estrutura física montada sobre a plataforma, como o cockpit com o painel frontal e o assento, embora apresente uma distribuição de massa aproximadamente uniforme em condição estacionária, pode gerar desbalanceamentos dinâmicos durante o movimento. Esses desbalanceamentos ocorrem devido à inércia das partes fixadas à base, especialmente quando há folgas estruturais ou diferentes rigidezes entre os componentes. Como consequência, surgem oscilações mecânicas, frequentemente percebidas como pequenos balanços residuais, que continuam a ser captados pelos sensores mesmo após a aparente estabilização da base.

Para analisar o impacto desses ruídos, duas abordagens complementares foram propostas. A primeira consistiu na comparação entre a resposta da plataforma utilizando o MCA e a mesma trajetória de movimento aplicada diretamente à base, apenas com escalonamento do sinal para se adequar ao espaço de trabalho da plataforma. Para garantir que os testes sejam idênticos, os dados de entrada foram previamente gravados em um arquivo rosbag, possibilitando a reprodução exata da trajetória para ambos os casos. A Figura 22 apresenta uma representação desse procedimento. Os sinais de *roll* e *pitch* foram coletados por meio de um acelerômetro,

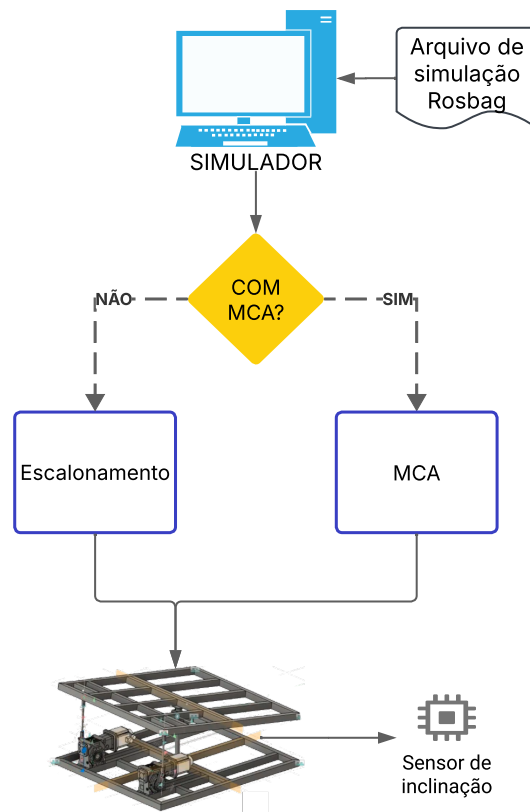


Figura 22 – Fluxograma de simulação em circuito

e filtrados para reduzir a influência do ruído proveniente do próprio sensor, isolando os sinais com o ruído da base.

Na segunda abordagem, buscou-se caracterizar o ruído estrutural do sistema em condições reais de operação. Foi selecionado um trecho de movimento em alta frequência, onde a ação do ruído se torna mais perceptível, e os sinais foram analisados sem aplicação de filtros, dado que o ruído dos sensores é constante entre os testes. O objetivo foi evidenciar o efeito das oscilações mecânicas, especialmente causadas por desbalanceamentos estruturais. A comparação entre os sinais foi realizada com base no desvio padrão, que fornece uma medida estatística da variação dos dados em torno de sua média, sendo uma métrica adequada para quantificar o nível de ruído (ANDRADE, 2020).

7 Resultados e Discussões

7.1 Validação experimental com o OMCT

Para a implementação do teste do OMCT em um sistema estacionário, foram aplicados individualmente sinais senoidais nas entradas correspondentes ao sistema. Com o objetivo de realizar uma análise detalhada e consistente da resposta em frequência, foram utilizados 12 sinais senoidais, de frequências variando entre 0,05 Hz e 1 Hz, limite superior escolhido pois, acima dessa frequência, o sistema apresenta instabilidade dinâmica crítica, com risco de comprometer a integridade estrutural da plataforma. A escolha da amplitude foi escolhida de magnitude 8 de aceleração translacional e velocidade angular. Embora a escolha não represente a capacidade máxima operacional do sistema, considerando que amplitudes mais elevadas, apesar de teoricamente desejáveis por proporcionarem uma melhor avaliação dos limites do sistema, introduzem efeitos indesejáveis. À medida que se aumenta a amplitude, surge uma tendência de maior desbalanceamento de massa no equipamento, o que, conseqüentemente, gera níveis superiores de ruído e vibração. Esse fenômeno poderia comprometer significativamente a qualidade dos dados coletados e, por conseqüência, a precisão das análises subsequentes, especialmente em situações em que a base do equipamento não possui ancoragem adequada ao solo.

Para cada frequência testada, foram efetuadas medições detalhadas do ganho e da diferença de fase entre os sinais de entrada e saída. Essas medições permitiram a elaboração dos gráficos de Bode correspondentes, os quais oferecem uma clara visualização da resposta dinâmica do sistema sob diferentes condições operacionais. Vargas (2009) utilizou um modelo de MCA clássico semelhante ao utilizado neste projeto em sua tese, porém em outra plataforma de movimentos, portanto para simulação, os parâmetros utilizados por Vargas (2009) foram implementados e comparados nos testes do OMCT com os modelos projetados especificamente para o MCA. Os resultados obtidos encontram-se ilustrados nas Figuras 23 e 24.

A Figura 23 apresenta o gráfico referente ao acoplamento direto do canal de rotação aplicado ao movimento *roll*. Já a Figura 24 exibe o gráfico correspondente ao mesmo tipo de acoplamento, mas aplicado ao movimento *pitch*. Ambos os gráficos indicam claramente a redução das baixas frequências devido à aplicação dos filtros no canal de rotação.

Considerando a simulação voltada para um caminhão fora de estrada, não é recomendável uma atenuação completa das baixas frequências, pois é necessário preservar parte do balanço natural do veículo durante a operação real. Idealmente,

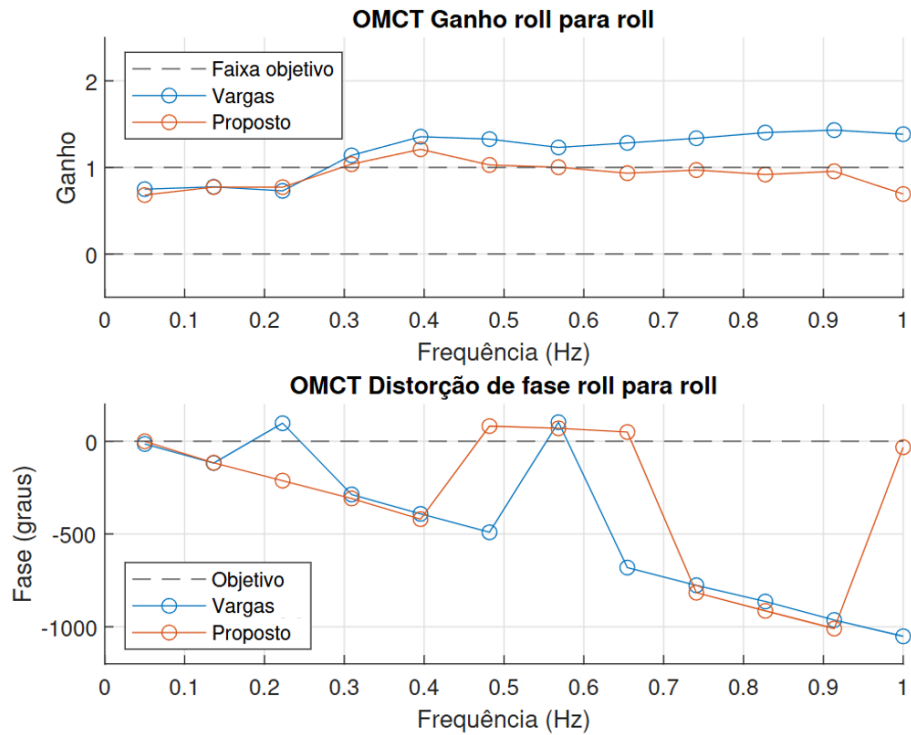


Figura 23 – Resposta em frequência entrada *roll* saída *roll*

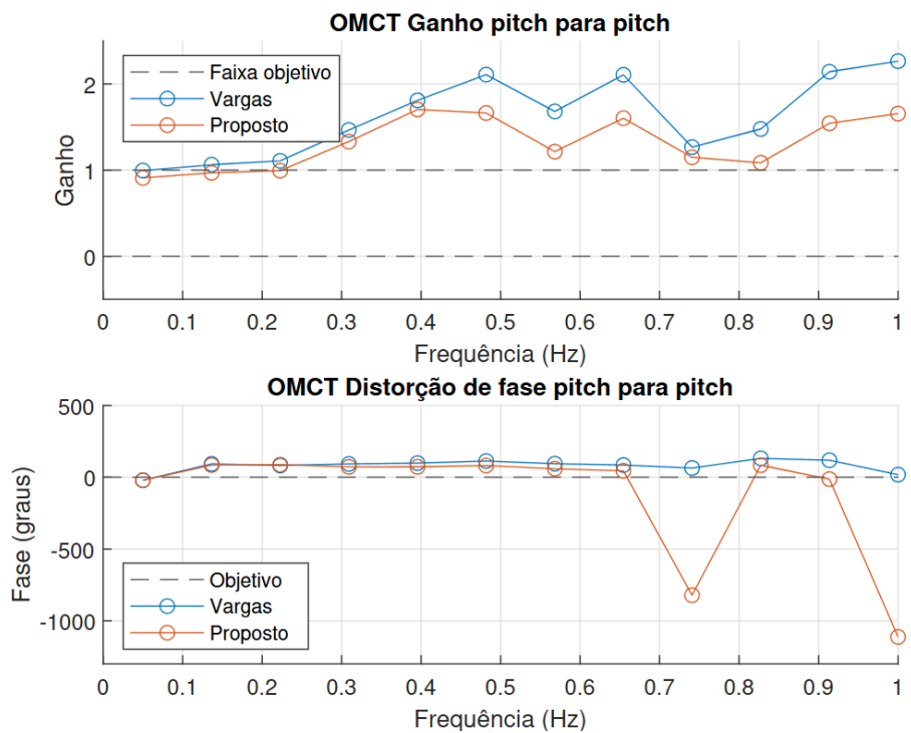


Figura 24 – Resposta em frequência entrada *pitch* saída *pitch*

busca-se alcançar ganhos próximos a 1 nas frequências mais elevadas, indicando uma boa fidelidade na reprodução dos movimentos. Além disso, espera-se uma mínima distorção de fase para evitar atrasos significativos na resposta dinâmica do sistema.

Ao examinar especificamente a Figura 23, percebe-se que o modelo proposto demonstra desempenho superior em comparação com os parâmetros adotados por Vargas (2009), mantendo ganhos consistentemente próximos de 1 em frequências mais altas, o que representa uma resposta de sensação de *roll* mais próxima do ambiente simulado. À medida que o efeito dos filtros se intensifica, torna-se mais evidente o aumento dos atrasos de fase, embora o modelo proposto continue mostrando menores atrasos e maior estabilidade em termos gerais. Parte dessa discrepância pode ser atribuída à maior instabilidade mecânica da base em frequências elevadas, o que potencializa oscilações indesejadas e pode atrasar a resposta do sistema. Além disso, durante a aquisição dos dados, foram observados indícios de pequenos atrasos ou perdas pontuais na captação pelo sensor, especialmente ao sair da inércia, o que também pode ter contribuído para os desvios registrados.

Na análise da Figura 24, é notável um ganho constante ao longo dos testes, possivelmente relacionado à presença de ruídos causados pelos componentes mecânicos da plataforma, sendo mais evidentes neste eixo de movimento. Apesar disso, o modelo sugerido apresenta melhor capacidade de replicação dos movimentos do simulador ao aproximar o sinal de saída da faixa objetivo. Embora ocorram picos pontuais na distorção de fase, ambos os métodos exibem comportamentos bastante semelhantes, o que representa um tempo de resposta semelhante entre eles.

Já em relação aos acoplamentos cruzados, para simular uma aceleração translacional constante de $8m/s^2$, por meio da inclinação da plataforma, seria necessário que a plataforma atinja um ângulo aproximado de 53° conforme a Equação 7.2. Entretanto, a plataforma é limitada a 20° , dessa forma, o ganho esperado do acoplamento cruzado é limitado a 2,5, como visto na Equação 7.3.

$$\theta = \arcsen\left(\frac{a}{g}\right) \quad (7.1)$$

$$\theta = \arcsen\left(\frac{8}{9,82}\right) = 53^\circ \quad (7.2)$$

$$Ganho = \frac{20}{8} = 2,5 \quad (7.3)$$

A redução nos ganhos dos acoplamentos cruzados, evidenciada nas Figuras 25 e 26, era prevista, dado que a limitação da taxa de inclinação a $3^\circ/s$ impede que a plataforma acompanhe a frequência de variações do teste, fazendo com que a base não consiga atingir a amplitude total desejada durante os experimentos.

Apesar dessa limitação, o método desenvolvido consegue reproduzir a sensação de movimento do simulador com precisão maior que a metodologia de Vargas (2009), especialmente na Figura 25. A principal vantagem do algoritmo proposto está na menor defasagem, evidenciada por uma margem de fase mais favorável, o que contribui para uma resposta mais ágil do sistema.

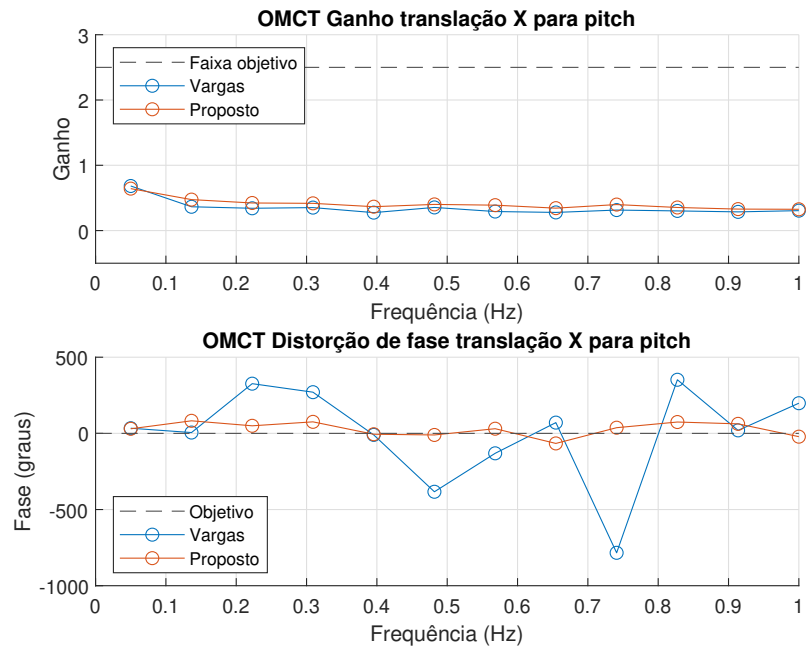


Figura 25 – Resposta em frequência entrada X saída *pitch*.

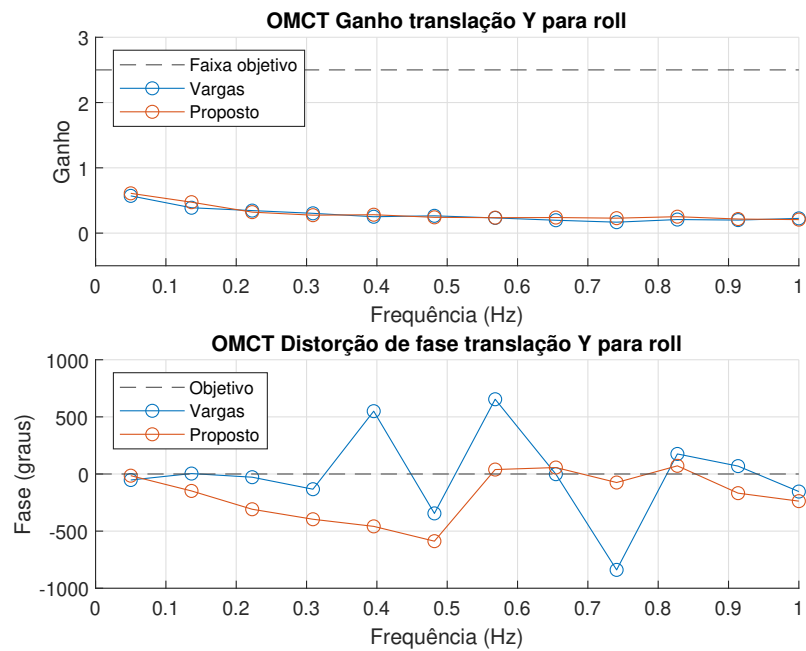


Figura 26 – Resposta em frequência entrada Y saída *roll*.

Observa-se que, em ambos os testes, o modelo proposto demonstra um desempenho superior, caracterizado por menores atrasos de fase e ganhos mais elevados, indicando uma resposta mais eficiente frente às limitações impostas pela dinâmica da plataforma.

7.2 Análise gráfica do movimento da plataforma

Após a execução do teste OMCT, foi realizada uma simulação virtual no ambiente do simulador, na qual um circuito contendo uma série de manobras representativas foi percorrido. Durante essa simulação, foram registrados os sinais correspondentes aos movimentos realizados, incluindo acelerações lineares e velocidades angulares, e salvos em um arquivo do tipo rosbag, possibilitando sua reutilização em testes posteriores.

Na etapa seguinte, foram conduzidos dois testes distintos utilizando a base de movimentos. No primeiro, os dados registrados no rosbag foram enviados diretamente à base. Durante esse teste, os sinais de *pitch* e *roll* gerados pela base foram capturados para análise.

No segundo teste, os mesmos dados do circuito foram processados pelo MCA proposto, o qual aplicou as estratégias de distribuição de movimento segundo o algoritmo desenvolvido. Os sinais gerados pelo MCA foram então enviados à base de movimentos, e novamente os sinais de *pitch* e *roll* foram registrados.

As Figuras 27 e 28 ilustram o comportamento da base de movimentos operando sob duas condições distintas: com e sem a aplicação do algoritmo MCA. Devido à extensão dos dados, são exibidas apenas uma janela de tempo visualmente adequada dos sinais capturados. Quando a base é acionada diretamente pelos sinais gerados pelo simulador, sem o uso do MCA, observa-se que os movimentos são reproduzidos de forma fiel, acompanhando diretamente a trajetória de referência. No entanto, esse modo de operação também evidencia a presença de ruídos nos sinais adquiridos, especialmente derivados da leitura do acelerômetro. Para mitigar esses efeitos indesejados, foi necessário aplicar filtros aos sinais, e as imagens apresentadas correspondem aos dados plotados após a aplicação desses filtros. Ainda assim, observam-se componentes de alta frequência residuais, indicando que a filtragem, embora eficaz, não elimina completamente essas interferências.

Além disso, nota-se que os sinais provenientes do simulador apresentam magnitudes mais elevadas em comparação com aqueles gerados pelo MCA. Essa diferença ocorre porque o MCA busca constantemente retornar a cadeira à posição neutra de forma imperceptível, após a aplicação de um movimento. Tal comportamento é intencional e desejável, pois o sistema vestibular humano é sensível a variações no movimento acima do limiar de transição (HUANG; FU, 2006). Dessa forma, a utilização do MCA permite que a plataforma opere de maneira mais eficiente dentro de um espaço de trabalho limitado, promovendo uma distribuição mais equilibrada dos movimentos.

A análise dos gráficos também revela que a resposta gerada pelo MCA difere significativamente do sinal original de *pitch* e *roll* registrado no simulador. Essa

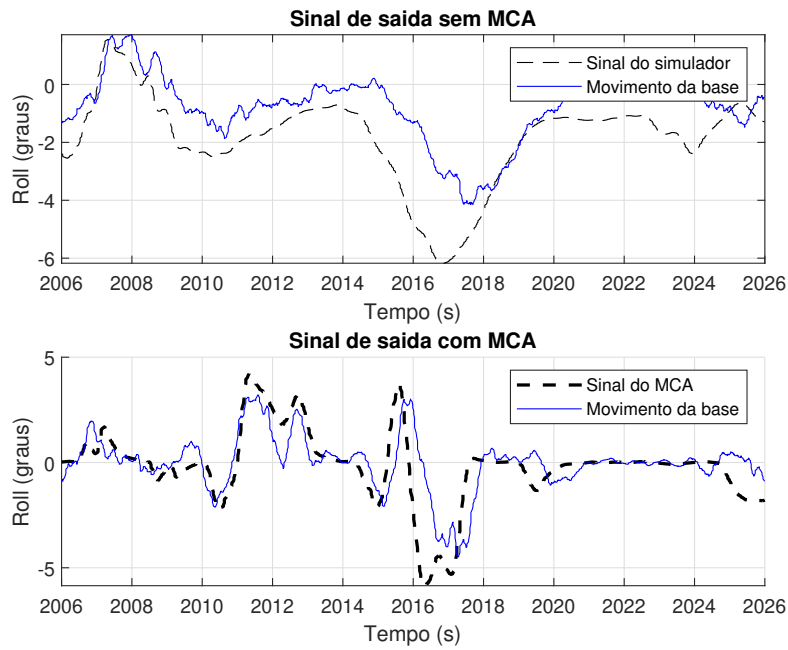


Figura 27 – Comparação da simulação com e sem o MCA - *roll*

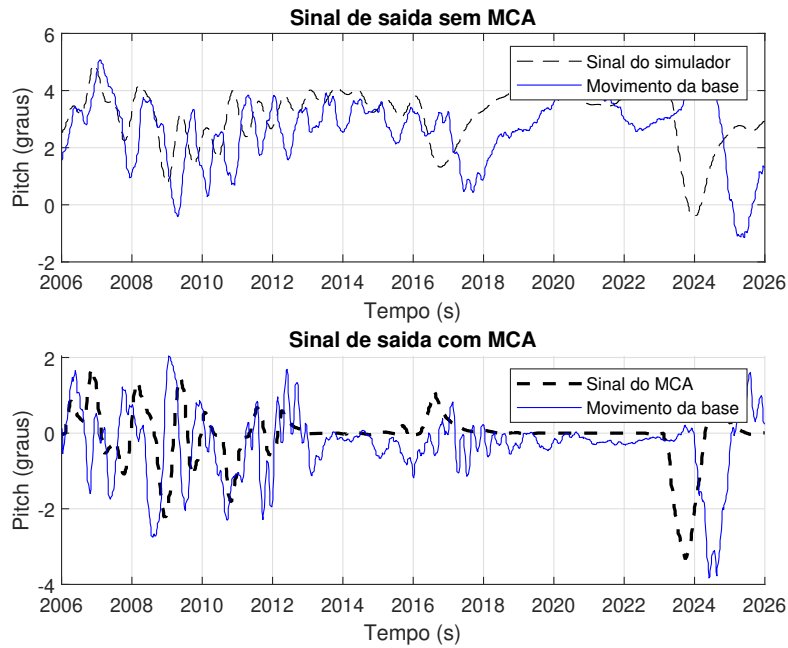


Figura 28 – Comparação da simulação com e sem o MCA - *pitch*

divergência está relacionada à atuação dos acoplamentos cruzados, os quais desempenham um papel central no algoritmo, convertendo acelerações translacionais em rotações equivalentes. Esse mecanismo é essencial para proporcionar uma experiência mais realista, onde os órgãos otólitos são capazes de compreender uma aceleração translacional por meio de uma inclinação da plataforma.

Por fim, observa-se um atraso na resposta da plataforma em ambas as condições. Parte desse atraso é atribuída às limitações na taxa de aquisição de dados do acelerômetro, enquanto no caso do MCA, o próprio tempo de processamento dos filtros e do algoritmo contribui para um acréscimo na defasagem temporal. Apesar disso, os resultados obtidos demonstram que o sistema é capaz de fornecer respostas coerentes com os objetivos da simulação, respeitando as limitações dinâmicas da base de movimento.

7.3 Avaliação da Resposta Vestibular ao Sinal do MCA

Para estimar a percepção do movimento gerada pelo sinal de saída do MCA, o algoritmo projetado neste estudo foi aplicado aos modelos dos canais semicirculares e dos otólitos, conforme a modelagem proposta por Houck, Telban e Cardullo (2005), mencionados na Seção 3. Dessa forma, obtém-se uma estimativa da sensação de movimento que seria percebida pelo sistema vestibular humano.

Os dados de angulação foram obtidos por meio de um acelerômetro instalado na plataforma. No entanto, observou-se atraso na aquisição dos sinais, além de possíveis perdas de amostras, causando distorções na leitura do sinal. Para tratar essas inconsistências e permitir uma análise mais precisa entre os sinais de entrada e saída, foi utilizado um algoritmo de comparação temporal (DTW). Esse método permite alinhar dois sinais temporais mesmo quando apresentam variações na velocidade ou defasagens ao longo do tempo, proporcionando uma correspondência mais robusta entre os dados. Com base no alinhamento obtido pelo DTW, foi calculado o erro quadrático médio (RMSE), refletindo a diferença ponto a ponto ao longo do tempo entre as trajetórias alinhadas. Além disso, o coeficiente de correlação (CC) foi utilizado para quantificar a similaridade estatística entre as variações dos dados após o alinhamento.

A simulação do MCA aplicada ao sistema vestibular permitiu observar como a resposta sensorial gerada pela plataforma de movimento real se aproxima da resposta estimada do sistema vestibular da simulação. Para fins de apresentação gráfica, foi selecionada uma janela temporal representativa que permitisse observar com clareza o comportamento dinâmico dos sinais. De forma geral, os sinais simulados seguem bem a dinâmica dos sinais reais, o que demonstra uma reprodução adequada dos estímulos vestibulares, tanto nos canais semicirculares quanto nos otolíticos. Os gráficos das Figuras 29 a 32 mostram os resultados obtidos.

Na Figura 29, por exemplo, observa-se que o sinal simulado acompanha com boa precisão a resposta angular esperada no eixo de *roll*. O valor de RMSE de 0,1061 e o coeficiente de correlação de 0,71623 confirmam essa proximidade entre os sinais, com variações bem sincronizadas ao longo do tempo.

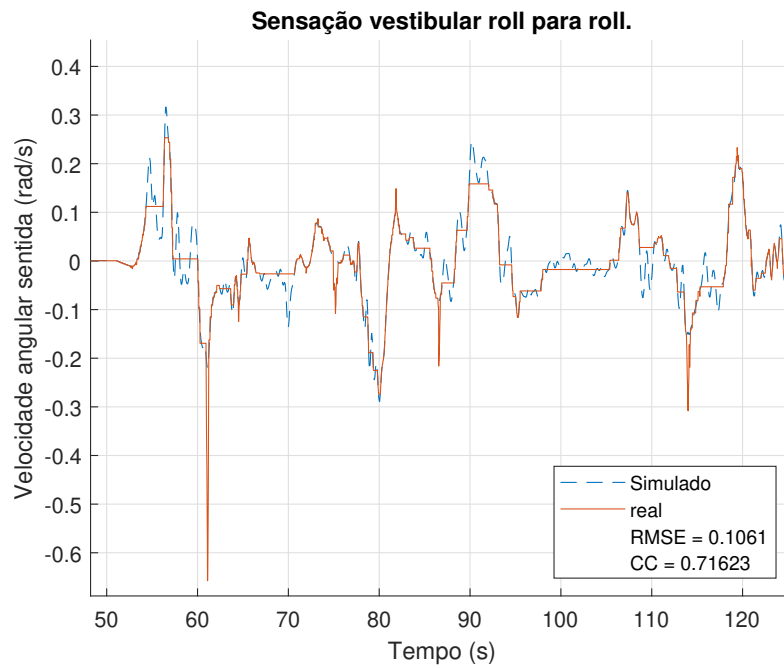


Figura 29 – Resposta vestibular da rotação *Roll* para *Roll*

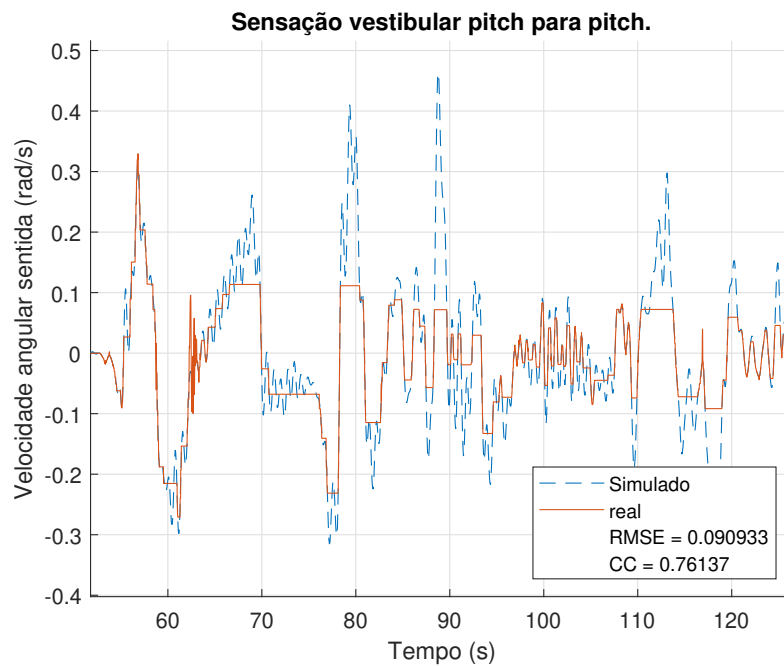


Figura 30 – Resposta vestibular da rotação *Pitch* para *Pitch*

A Figura (30) representa a resposta dos canais semicirculares no eixo de rotação frontal (*pitch*). Embora o RMSE (0,090933) e o CC (0,76137) indiquem uma alta similaridade entre os sinais, é possível observar uma atenuação na amplitude do sinal simulado em relação ao real. Tal comportamento pode ser atribuído a limitações mecânicas da plataforma de movimento, que possui alcance e aceleração angular

limitados, especialmente durante transições rápidas.

Os dados dos acoplamentos cruzados são representados nas Figuras 31 e 32. Estes movimentos de entrada translacional são sentidos pelos canais otolíticos. Estes canais apresentaram uma correlação menor e um erro mais alto, já que ocorre uma transformação de movimento neste canal.

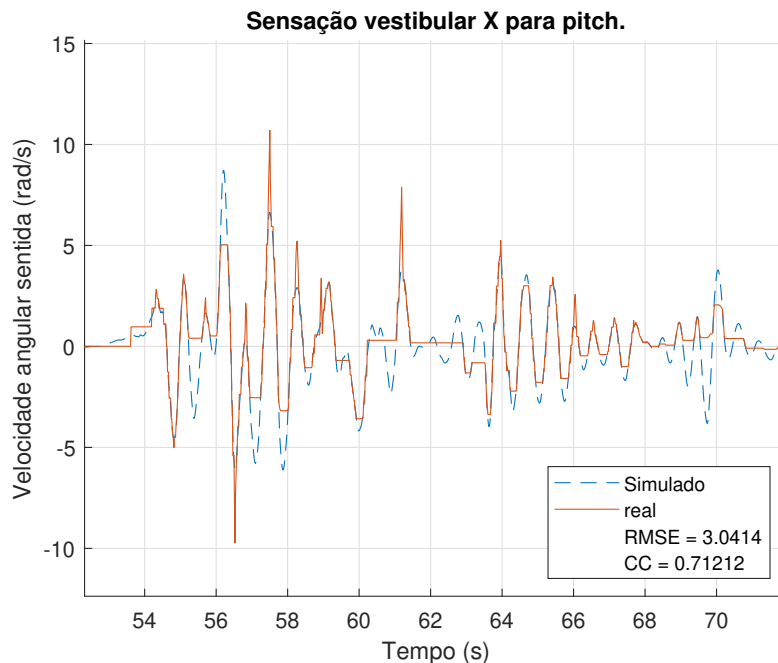


Figura 31 – Resposta vestibular da translação X para *Pitch*.

Para o eixo longitudinal (*surge*), o RMSE foi de 3,0414 e o coeficiente de correlação de 0,71212. Embora os sinais apresentem um alinhamento geral no padrão de variação, observa-se que, em determinados trechos, a resposta simulada ultrapassa os valores esperados, o que pode indicar a ocorrência de sensações intensificadas de movimento. Esse comportamento também é perceptível nas Figuras 29 e 32. Tais discrepâncias tendem a ocorrer em regiões com variações rápidas de aceleração, ou seja, onde há derivadas elevadas da posição, nas quais a plataforma pode gerar picos não compensados adequadamente pelo algoritmo. Esse fenômeno também pode se manifestar no início ou término de certos movimentos, quando a resposta inercial da plataforma é elevada, gerando oscilações associadas a desequilíbrios de massa já discutidos anteriormente.

Já na Figura 32, que corresponde ao eixo lateral dos otólitos (*sway*), com um RMSE de 1,8296 e CC de 0,73806, observa-se fenômeno semelhante ao da Figura 30. A atenuação do sinal em relação ao percebido na simulação é perceptível, o que indica que o sistema pode ter enfrentado certa limitação ao reproduzir acelerações mais rápidas e de curta duração. Esse comportamento pode estar relacionado tanto às restrições dinâmicas da plataforma, como alcance, velocidade e tempo de resposta.

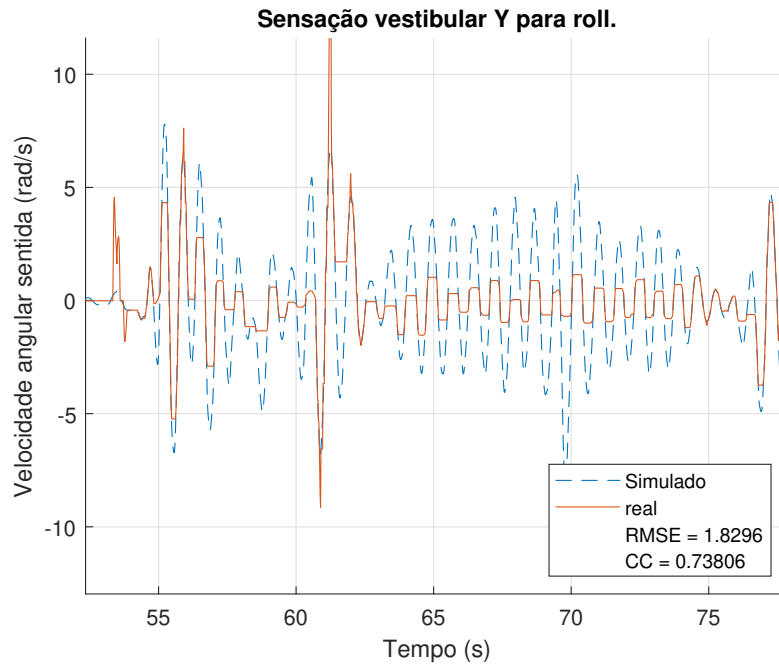


Figura 32 – Resposta vestibular da translação Y para *Roll*.

De modo geral, os resultados sugerem que o algoritmo MCA, mesmo diante das limitações da plataforma de movimento, é capaz de gerar respostas vestibulares reais compatíveis com aquelas estimadas a partir de um movimento do simulador de caminhões.

7.4 Análise do ruído no algoritmo de sensação de movimento

Com o objetivo de analisar o comportamento do sistema frente à presença de ruído, foi realizada uma avaliação específica utilizando uma entrada senoidal de frequência elevada. Essa escolha visa acentuar os efeitos do ruído, uma vez que oscilações indesejadas tornam-se mais perceptíveis nesse regime de operação. O sinal de *pitch* foi selecionado para essa análise por ser o eixo em que os efeitos do desbalanceamento de massa, ocasionado pelas estruturas fixadas sobre a plataforma, se manifestam com maior intensidade, resultando em perturbações significativas durante o movimento.

O teste foi conduzido em duas condições distintas: com a aplicação do algoritmo MCA e sem sua utilização. Em ambos os casos, os sinais resultantes foram capturados ao longo de um mesmo intervalo de tempo, permitindo uma comparação direta do nível de ruído presente nas respostas da plataforma. A Figura 33 apresenta os resultados obtidos, evidenciando a diferença entre o sinal processado pelo MCA e o sinal bruto.

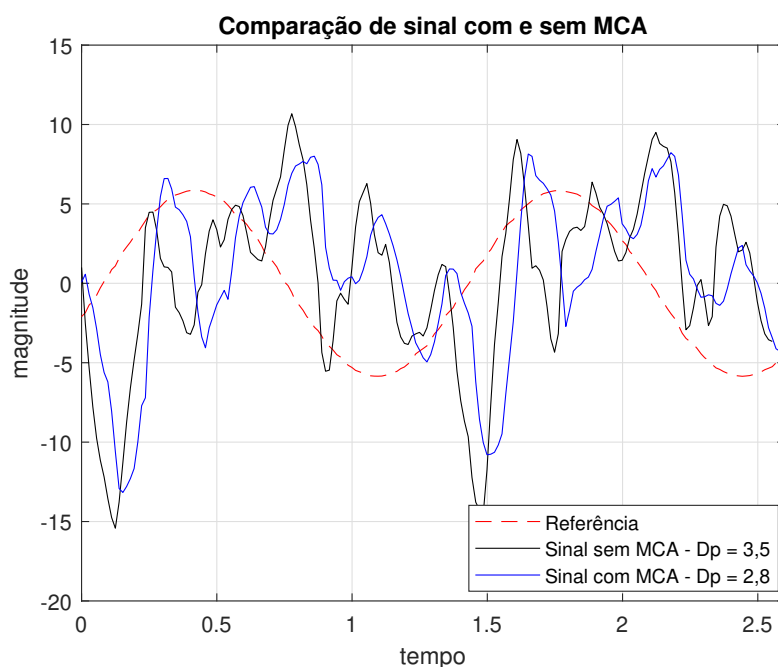


Figura 33 – Comparação de ruído do sistema com e sem MCA

Observa-se que, na ausência do MCA, o sinal de *pitch* apresenta oscilações de maior amplitude, característica típica da presença de ruído mecânico. Esse comportamento pode ser atribuído, principalmente, ao desbalanceamento das massas fixadas à estrutura móvel da base, que se torna mais perceptível em frequências mais altas. Por outro lado, quando o MCA é aplicado, nota-se uma atenuação significativa dessas oscilações, o que indica uma filtragem mais eficaz do ruído e uma aproximação maior ao comportamento desejado.

A redução do ruído foi quantificada por meio do cálculo do desvio padrão (D_p) dos sinais analisados. O sinal obtido sem o uso do MCA apresentou um desvio padrão de 3,5, enquanto o sinal processado pelo MCA obteve um valor reduzido de 2,8. Essa diferença reforça a eficácia do algoritmo em suavizar as perturbações presentes na resposta da base de movimento, promovendo uma operação mais estável e condizente com os objetivos do sistema de simulação.

8 Conclusão

Este trabalho apresentou o desenvolvimento e validação de um algoritmo de sensação de movimento aplicado a um sistema de simulação de caminhão fora de estrada, desenvolvido na Universidade Federal de Itajubá – Unifei, campus Itabira, em parceria com a empresa Vale S/A. A plataforma foi construída com dois graus de liberdade (2DoF), projetada para suportar até 500 kg, com o objetivo de atender às necessidades específicas de treinamento de operadores em ambientes de mineração, proporcionando uma experiência de simulação realista e segura.

O algoritmo de sensação de movimento adotado foi o filtro clássico de *washout*, uma abordagem amplamente utilizada em simuladores de movimento devido à sua estrutura simples, boa eficiência computacional e reconhecimento consolidado na literatura técnica. Sua escolha se deu também pela compatibilidade com plataformas que possuem limitações mecânicas mais severas, como a base desenvolvida neste estudo.

Durante os ensaios experimentais, foram percebidos ruídos de alta frequência, especialmente no sinal de inclinação (*pitch*). Esses ruídos estão associados à montagem da estrutura do cockpit do CAT 793F diretamente sobre a base de movimento, o que ocasiona maior desbalanceamento dinâmico e, conseqüentemente, maior oscilação durante movimentos com amplitudes elevadas. Essa maior suscetibilidade no eixo de inclinação pode estar relacionada à distribuição de massa da estrutura, que tende a acentuar o movimento nesse canal específico. Além disso, a base de movimento não estava rigidamente fixada ao solo, o que pode ter contribuído para pequenas oscilações indesejadas, fazendo com que a estrutura se deslocasse levemente do chão em determinados momentos.

A nova parametrização do algoritmo clássico proposta neste estudo foi comparada com a apresentada por Vargas (2009), utilizando o OMCT como critério de avaliação. Os resultados demonstraram que o modelo proposto apresentou melhorias significativas nos acoplamentos diretos (canal de rotação), onde os movimentos de *roll* e *pitch* possuem maior proximidade com os da realidade. Em relação aos acoplamentos cruzados, observou-se uma resposta mais fiel no eixo de *pitch* induzido pela translação no eixo X, enquanto o movimento de *roll* gerado pela translação em Y mostrou resultados semelhantes entre os modelos, mas com resposta mais rápida no modelo proposto. Essa proximidade nos resultados é limitada pela restrição da taxa angular de 3 graus por segundo, adotada para preservar o conforto do operador durante a operação.

Durante o processo de validação, foi também avaliado o desempenho do algoritmo

a partir da perspectiva fisiológica da sensação de movimento, por meio da simulação do sistema vestibular humano. As acelerações lineares e velocidades angulares registradas na plataforma foram processadas por modelos representativos dos canais otolíticos e semicirculares. Para viabilizar a comparação entre o movimento ideal (simulação do caminhão) e o movimento gerado pela base com o MCA, foi aplicado o algoritmo DTW, a fim de alinhar temporalmente os sinais e corrigir distorções causadas por atrasos e ruídos. Em seguida, foram calculados o erro quadrático médio (RMSE) e o coeficiente de correlação entre os sinais vestibulares.

Os resultados mostraram boa similaridade entre os sinais simulados e os obtidos na plataforma, especialmente nos canais semicirculares, que apresentaram elevada correlação e baixo erro. Já os canais otolíticos, responsáveis por detectar acelerações lineares, apresentaram respostas levemente atenuadas, possivelmente devido às limitações mecânicas da base e à necessidade de equilibrar a fidelidade do movimento com o conforto do operador. Pequenas discrepâncias observadas, como atenuações ou amplificações pontuais, são esperadas em ambientes com alto grau de dinamismo e complexidade, e oferecem oportunidades para ajustes e refinamentos futuros tanto no algoritmo quanto no sistema de controle da plataforma.

Como perspectivas para trabalhos futuros, recomenda-se a realização de estudos subjetivos com diferentes operadores, buscando avaliar a suavização e a qualidade dos movimentos do MCA, principalmente no canal de inclinação. Nesse contexto, propõe-se também a aplicação do teste OMCT para análise de acoplamento cruzado, utilizando amplitudes que respeitem os limites físicos da base de movimento, evitando possíveis atenuações no resultado. Também se sugere a comparação do filtro clássico de *washout* com outras abordagens modernas, como algoritmos baseados em otimização, redes neurais ou estratégias adaptativas, visando aprimorar ainda mais a qualidade da sensação de movimento gerada. Além disso, destaca-se a importância de investigar métodos de fixação da base ao solo, com o intuito de eliminar os efeitos de deslocamento observados durante os movimentos mais intensos.

Por fim, observa-se que a aplicação do algoritmo clássico de *washout* em plataformas com características construtivas similares à adotada neste estudo ainda é pouco explorada na literatura. Sendo assim, futuras comparações com sistemas semelhantes podem contribuir para situar melhor o presente trabalho dentro do cenário acadêmico, posicionando-o como uma referência contextual importante e como uma ponte entre pesquisas científicas e aplicações industriais voltadas à capacitação de operadores em ambientes críticos.

Referências

ALINA-ELENA, L. M.; GHEORGHE, G. Software tools used for research and prototyping a gough-stewart platform. *Engineering Series*, n. 2, 2023. Citado na página 18.

ANDRADE, C. Understanding statistical noise in research: 1. basic concepts. *Indian J Psychol Med.*, v. 45, n. 1, p. 89–90, 2020. Citado na página 45.

ANGELAKI, D. E.; CULLEN, K. E. Vestibular system: The many facets of a multimodal sense. *Annual Review of Neuroscience*, Annual Reviews, v. 31, n. Volume 31, 2008, p. 125–150, 2008. Citado 2 vezes nas páginas 27 e 28.

ARANGO, C. G. et al. Learning-based approximate nonlinear model predictive control motion cueing. *arXiv e-prints*, p. arXiv–2504, 2025. Citado na página 35.

ARIOUI, H.; NEHAOUA, L.; AMOURI, H. Classic and adaptive washout comparison for a low cost driving simulator. In: IEEE. *Proceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation Intelligent Control, 2005*. [S.l.], 2005. p. 586–591. Citado 2 vezes nas páginas 19 e 30.

ASADI, H. et al. A review on otolith models in human perception. *Behavioural brain research*, Elsevier, v. 309, p. 67–76, 2016. Citado na página 24.

ASADI, H. et al. Optimisation of nonlinear motion cueing algorithm based on genetic algorithm. *Vehicle System Dynamics*, Taylor & Francis, v. 53, n. 4, p. 526–545, 2015. Citado na página 22.

ASADI, H. et al. A particle swarm optimization-based washout filter for improving simulator motion fidelity. In: IEEE. *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. [S.l.], 2016. p. 001963–001968. Citado na página 21.

BAZONI, J. A. *Hipovitaminose D como fator de risco para a vertigem posicional paroxística benigna: estudo caso-controle*. Dissertação (Mestrado) — Universidade Norte do Paraná, 2016. Citado 3 vezes nas páginas 8, 27 e 28.

BEZERRIL, N.; OLIVEIRA, F. A influência do sistema vestibular na manutenção do equilíbrio corporal. In: . [S.l.]: Universidade Federal da Paraíba, 2022. Citado 2 vezes nas páginas 26 e 28.

BOHIGAS, O.; MANUBENS, M.; ROS, L. A linear relaxation method for computing workspace slices of the stewart platform. *Journal of Mechanisms and Robotics*, v. 5, p. 011005, 10 2012. Citado 2 vezes nas páginas 8 e 19.

CASAS-YRURZUM, S. et al. On the objective evaluation of motion cueing in vehicle simulations. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 22, n. 5, p. 3001–3013, 2020. Citado 2 vezes nas páginas 24 e 41.

CAT. *Caminhões de Mineração*. <https://www.cat.com/pt_BR/products/new/equipment/off-highway-trucks/mining-trucks/18092621.html> [Acessado em: (27/06/2025)]. Citado na página 13.

CELESTINO, M. S.; VALENTE, V. C. P. N. Aplicabilidade e benefícios de softwares e simuladores em processos de ensino-aprendizagem. *ETD - Educação Temática Digital*, v. 23, n. 4, p. 881–903, nov. 2021. Citado na página 13.

CONRAD, B.; SCHMIDT, S. F. *Motion drive signals for piloted flight simulators*. [S.l.], 1970. Citado 2 vezes nas páginas 12 e 21.

DELGADO, L. C. *Design and assembly of a 2-DOF motion simulator for racing applications*. Dissertação (B.S. thesis) — Universitat Politècnica de Catalunya, 2025. Citado na página 13.

FUENTES, J. *Motion cueing study and development for driving simulators*. Dissertação (Tesi di Laurea Magistrale) — Politecnico di Milano, 2021. Citado na página 21.

GARRETT, N. J.; BEST, M. Driving simulator motion cueing algorithms—a survey of the state of the art. Loughborough University, 2010. Citado na página 13.

GRANT, P. R.; REID, L. D. Motion washout filter tuning: Rules and requirements. *Journal of aircraft*, v. 34, n. 2, p. 145–151, 1997. Citado na página 12.

HOSMAN, R.; ADVANI, S. Design and evaluation of the objective motion cueing test and criterion. *The Aeronautical Journal*, Cambridge University Press, v. 120, n. 1227, p. 873–891, 2016. Citado na página 25.

HOUCK, J.; TELBAN, R.; CARDULLO, F. Motion cueing algorithm development: Human-centered linear and nonlinear approaches. 06 2005. Citado 6 vezes nas páginas 25, 27, 28, 29, 43 e 52.

HUANG, C.-I.; FU, L.-C. Human vestibular based (hvb) senseless maneuver optimal washout filter design for vr-based motion simulator. In: IEEE. *2006 IEEE International Conference on Systems, Man and Cybernetics*. [S.l.], 2006. v. 5, p. 4451–4458. Citado na página 50.

KARMAKAR, S.; TURNER, C. J. Forward kinematics solution for a general stewart platform through iteration based simulation. *The International Journal of Advanced Manufacturing Technology*, Springer, v. 126, n. 1, p. 813–825, 2023. Citado na página 19.

KHUSRO, Y. R. et al. Mpc-based motion-cueing algorithm for a 6-dof driving simulator with actuator constraints. *Vehicles*, MDPI, v. 2, n. 4, p. 625–647, 2020. Citado na página 22.

KRAG, B. Modeling and simulation—basics and benefits. *In-Flight Simulators and Fly-by-Wire/Light Demonstrators: A Historical Account of International Aeronautical Research*, Springer, p. 19–25, 2017. Citado na página 17.

LEMES, R. C. Desenvolvimento de um algoritmo de movimento adaptativo fuzzy-washout clássico para uma plataforma de stewart. 2018. Citado na página 35.

LIU, C.; LI, L.; ZHAO, L. A motion cueing algorithm for a 2-dof pneumatic motion simulation platform. In: IEEE. *2023 42nd Chinese Control Conference (CCC)*. [S.l.], 2023. p. 455–460. Citado na página 19.

MACENSKI, S. et al. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, v. 7, n. 66, p. eabm6074, 2022. Disponível em: <<https://www.science.org/doi/abs/10.1126/scirobotics.abm6074>>. Citado na página 32.

MERLET, J.-P. *Parallel robots*. [S.l.]: Springer Science & Business Media, 2005. v. 128. Citado na página 30.

MURGOVSKI, N. Vehicle modelling and washout filter tuning for the chalmers vehicle simulator. 2007. Citado na página 23.

NEHAOUA, L. et al. Motion cueing algorithms for small driving simulator. In: IEEE. *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006*. [S.l.], 2006. p. 3189–3194. Citado 2 vezes nas páginas 19 e 22.

PEREIRA, M. *Labirintite, você sabe o que é?* [S.l.]: Universidade de Brasília, 2021. <<https://www.citedrive.com/overleafhttps://eupercebo.unb.br/2021/11/08/labirintite-voce-sabe-o-que-e/>> [Acessado em: (21/05/2025)]. Citado 2 vezes nas páginas 8 e 26.

QAZANI, M. R. C. et al. An mpc-based motion cueing algorithm using washout speed and grey wolf optimizer. In: IEEE. *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. [S.l.], 2021. p. 1627–1633. Citado na página 22.

QAZANI, M. R. C. et al. Adaptive motion cueing algorithm based on fuzzy logic using online dexterity and direction monitoring. *IEEE Systems Journal*, IEEE, v. 16, n. 2, p. 1945–1953, 2021. Citado 2 vezes nas páginas 23 e 35.

QAZANI, M. R. C. et al. A neural network-based motion cueing algorithm using the classical washout filter for comprehensive driving scenarios. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 25, n. 6, p. 5112–5121, 2023. Citado na página 22.

RANAWEERA, M.; MAHMOUD, Q. H. Bridging reality gap between virtual and physical robot through domain randomization and induced noise. In: *Canadian AI*. [S.l.: s.n.], 2022. Citado na página 44.

RENGIFO, C. et al. Impact of human-centered vestibular system model for motion control in a driving simulator. *IEEE Transactions on human-machine systems*, IEEE, v. 51, n. 5, p. 411–420, 2021. Citado 2 vezes nas páginas 15 e 24.

REYMOND, G.; KEMENY, A. Motion cueing in the renault driving simulator. *Vehicle System Dynamics*, Taylor & Francis, v. 34, n. 4, p. 249–259, 2000. Citado na página 24.

SANTOS, V. C. dos; SILVEIRA, G. A. da. A efetividade dos simuladores de voo no treinamento de pilotos para tarefas processuais específicas e aquisição de habilidades. *Revista Conexão SIPAER*, v. 10, n. 1, p. 15–30, 2019. Citado na página 12.

SEEHOF, C.; DURAK, U.; DUDA, H. Objective motion cueing test-experiences of a new user. In: *AIAA Modeling and Simulation Technologies Conference*. [S.l.: s.n.], 2014. p. 2205. Citado 3 vezes nas páginas 15, 25 e 41.

SPRINGER, K.; GATTRINGER, H.; BREMER, H. Towards washout filter concepts for motion simulators on the base of a stewart platform. *PAMM*, Wiley Online Library, v. 11, n. 1, p. 955–956, 2011. Citado na página 36.

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 na página 17.

STUPANS, A. et al. Implementation of classical motion cueing algorithms for serial manipulator based motion simulator: Larrmos case study. *Latvian Journal of Physics and Technical Sciences*, De Gruyter Poland, v. 61, n. 4, p. 3–16, 2024. Citado 3 vezes nas páginas 12, 21 e 35.

TONG, P.; BIL, C. Effects of errors on transfer of aircraft flight training. *National Aerospace Laboratory NLR - 32nd European Rotorcraft Forum, ERF 2006*, v. 3, p. 1390–1411, 01 2007. Citado 2 vezes nas páginas 8 e 18.

ULLRICH, W. The antoinette barrel – a training device upon request. In: . [S.l.: s.n.], 2022. <<https://www.halldale.com/defence/20498-the-antoinette-barrel-a-training-device-upon-request>> [Acessado em: (02/06/2025)]. Citado 2 vezes nas páginas 8 e 18.

VARGAS, M. B. *Controle de uma Plataforma de Movimento de um Simulador de voo*. Tese (Doutorado) — Universidade de São Paulo, 2009. Citado 8 vezes nas páginas 8, 36, 37, 39, 40, 46, 48 e 57.

YADAV, M.; ALAM, M. A. Dynamic time warping (dtw) algorithm in speech: a review. *International Journal of Research in Electronics and Computer Engineering*, v. 6, n. 1, p. 524–528, 2018. Citado na página 43.

YOUNG, L.; OMAN, C. Model for vestibular adaptation to horizontal rotation. *Aerospace medicine*, v. 40, n. 10, p. 1076–1080, 1969. Citado na página 25.

ZHANG, C.; ZHANG, L. Kinematics analysis and workspace investigation of a novel 2-dof parallel manipulator applied in vehicle driving simulator. *Robotics and Computer-Integrated Manufacturing*, v. 29, n. 4, p. 113–120, 2013. ISSN 0736-5845. Citado 4 vezes nas páginas 8, 20, 30 e 31.

ZHAO, J. et al. Six-dof stewart platform motion simulator control using switchable model predictive control. *arXiv preprint arXiv:2503.11300*, 2025. Citado 2 vezes nas páginas 18 e 35.

A APÊNDICE

A.1 Código em Python de funções utilizadas

```
1  import scipy.signal as signal
2
3  # Função para calcular equação característica
4  def resultado_FZ(FZ, u, y):
5      # Entrar com formato FS=[num,dem] y=[y(n-1),y(n-2),...] u=[u(n),u(n-1),...]
6      # y tem uma informação a mais do que o u, para nivelar é feito isto
7      y = y[:-1]
8      num = FZ[0]
9      den = FZ[1]
10     vy = 0
11     vu = 0
12
13     if len(y) < len(den) - 1:
14         print("erro: Quantidade de instantes de y nao e suficiente")
15     if len(u) < len(num):
16         print("erro: Quantidade de instantes de u nao e suficiente")
17
18     for i in range(1, len(den)):
19         vy = vy - den[i] * y[i - 1]
20
21     for i in range(len(num)):
22         vu = vu + num[i] * u[i]
23
24     Y_out = vy + vu
25     return Y_out
26
27 # calcula derivada de u e retorna um vetor com último valor de cada sinal
28 def deriva(u, tempo_amostragem):
29     import numpy as np
30
31     u= u.T
32     nlinhas = len(u)
33     valor = np.full(nlinhas, 0)
34
35     for i in range(nlinhas):
36         derivada = np.diff(np.flip(u[i])) / tempo_amostragem
37         valor[i] = derivada[-1]
38
39     return valor
40
```

```

41 #integra
42 def integra(u, tempo_amostragem):
43     import numpy as np
44
45     # Calcular a integral utilizando a regra do trapézio
46     integral = np.trapz(u, dx=tempo_amostragem)
47
48     return integral
49
50 def cria_funcao(FS, Ts):
51     # Converta a função de transferência para tempo discreto
52     num, den, _ = signal.cont2discrete(FS, Ts, method="tustin")
53     num = num[0]
54
55
56     Fz = [num, den]
57     return Fz
58
59 #define taxa de variação de saída
60 def limite_rater(v_atual, v_ant, max_rate):
61     if abs(v_atual - v_ant) < max_rate:
62         return v_atual
63     else:
64         if v_atual > v_ant:
65             resultado = v_ant + max_rate
66         else:
67             resultado = v_ant - max_rate
68
69     return resultado
70
71
72 # Gera matriz de rotação R e matriz de taxa de variação angular Rw
73 def matriz_rotacao(angulos):
74     import numpy as np
75     import math
76
77     roll = angulos[0]
78     pitch = angulos[1]
79     yaw = angulos[2]
80
81     Rx = [
82         [1, 0, 0],
83         [0, math.cos(roll), -(math.sin(roll))],
84         [0, math.sin(roll), math.cos(roll)],
85     ]
86     Ry = [
87         [math.cos(pitch), 0, math.sin(pitch)],
88         [0, 1, 0],

```

```

89     [-(math.sin(pitch)), 0, math.cos(pitch)],
90     ]
91     Rz = [
92         [math.cos(yaw), -(math.sin(yaw)), 0],
93         [math.sin(yaw), math.cos(yaw), 0],
94         [0, 0, 1],
95     ]
96
97     # Rz * Ry * Rx
98     R = np.dot(np.dot(Rz, Ry), Rx)
99
100    Rw = [
101        [math.cos(yaw) * math.cos(pitch), -(math.sin(yaw)), 0],
102        [math.cos(pitch) * math.sin(yaw), math.cos(yaw), 0],
103        [-(math.sin(yaw)), 0, 1],
104    ]
105
106    return [R, Rw]

```

A.2 Código em Python do MCA

```

1
2
3 import numpy as np
4 from my_package import funcoes as func
5 import math
6 import rclpy
7 from rclpy.node import Node
8 import time
9 from std_msgs.msg import Float32MultiArray
10
11 #PARÂMETROS -----
12
13 OMCT = True
14 topico_lido = "/motionsystem"
15 topico_escrito = '/out_MCA'
16 #-----
17 class ControlSystemNode(Node):
18     def __init__(self):
19
20         super().__init__("control_system_node")
21         self._new_information_subscriber = self.create_subscription(
22             Float32MultiArray, topico_lido,
23             self._new_information_callback, 10
24         )

```

```

25
26         # Publisher
27         self.publisher_ = self.create_publisher(Float32MultiArray,
28         topico_escrito, 10)
29
30         # Atributos do MCA
31         self._staturation_top_block = 1 #! Review value
32         self._gain_translational_acceleration = 2
33         self._gain_angular_velocity = 2
34         self._deriv2_limiter = 1000 #Valor máximo da segunda derivada
35         self._g = np.array([0, 0, 9.81])
36         self._r_matrix_R = np.zeros((3, 3))
37         self._rw_matrix_R = np.zeros((3, 3))
38         self.Z_center = 1 #posição na altura z
39         self._taxa = 0.0524 # limite de variação de percepção
40         self._Ts = 0.05 # Tempo de amostragem
41         self.escrita = Float32MultiArray()
42
43         # Funções de transferência
44         FS_translacaoZ = [[1, 0, 0], [1, 6, 16, 0, 0]]
45         FS_translacaoXY = [[1, 0, 0], [1, 5, 6.25, 0, 0]]
46         FS_inclinacaoY = [[3.5*3.5], [1, 2*3.5*0.7, 3.5*3.5]]
47         FS_inclinacaoX = [[9], [1, 0.7*3*2, 9]]
48         FS_rotacaoZ = [[1, 0], [1, 1, 0]]
49         FS_rotacaoXY = [[1, 0], [1, 1, 0]]
50
51         #numeradores e denominadores das equações de diferenças
52         self._FZ_translacaoXY = func.cria_funcao(
53         FS_translacaoXY, self._Ts)
54         self._FZ_translacaoZ = func.cria_funcao(FS_translacaoZ, self._Ts)
55         self._FZ_inclinacaoX = func.cria_funcao(FS_inclinacaoX, self._Ts)
56         self._FZ_inclinacaoY = func.cria_funcao(FS_inclinacaoY, self._Ts)
57         self._FZ_rotacaoXY = func.cria_funcao(FS_rotacaoXY, self._Ts)
58         self._FZ_rotacaoZ = func.cria_funcao(FS_rotacaoZ, self._Ts)
59
60         self.sinal_OMCT = [0.0,0.0,0.0,0.0,0.0,0.0]
61
62         self.t=0
63         self.tplot =[]
64         self.yplot =[]
65
66         #Matriz principal que armazena ate 5 tempos anteriores
67         titulo = {
68             "u",
69             "y",
70             "u_derivado",
71             "u_translacaoX",
72             "u_translacaoY",

```

```

73         "u_translacaoZ",
74         "y_translacaoX",
75         "y_translacaoY",
76         "y_translacaoZ",
77         "u_inclinacaoX",
78         "u_inclinacaoY",
79         "y_inclinacaoX",
80         "y_inclinacaoY",
81         "u_rotacaoX",
82         "u_rotacaoY",
83         "u_rotacaoZ",
84         "y_rotacaoX",
85         "y_rotacaoY",
86         "y_rotacaoZ",
87         "tiltX",
88         "tiltY",
89     }
90
91     self._matriz_sinais = {}
92     # cria uma matriz de 5 posições com os tempos anteriores de
93     #'titulo' inicializado com valor 0.000001
94     # formato do vetor é 'y' = [[Yx Yy Yz Yroll Ypith Yyaw]
95     # [Yx(n-1) Yy(n-1) Yz(n-1) Yroll(n-1) Ypith(n-1) Yyaw(n-1)]...]
96     for nome in titulo:
97         self._matriz_sinais[nome] = np.full(5, 0.000001,
98         dtype=object)
99
100     # inicializa as instâncias de u com 6 elementos
101     self._matriz_sinais["u"] = [
102     np.full(6, 0.000001) for _ in range(5)]
103     self._matriz_sinais["u_derivado"] = \
104     [np.full(6, 0.000001) for _ in range(5)]
105
106     self.atualiza_matriz("u", [0,0,0,0,0,0])
107
108     self.publisher_.publish(self.escrita)
109     time.sleep(3)
110     #CALLBACK (MAIN)-----
111
112     def _new_information_callback(self, msg):
113
114         self.atualiza_matriz("u", msg.data)
115
116         #Função que captura saída do canal de translação e
117         #entrada do canal de inclinação
118         translational_channel_output, slop_channel_output = \
119         self.top_block()
120

```

```

121         #Função que captura saída do canal de rotação
122         bottom_block_result = self.bottom_block(slop_channel_output)
123
124         self.escrita.data = translational_channel_output+ \
125         bottom_block_result.tolist()
126         self.publisher_.publish(self.escrita)
127
128     # ----- FUNÇÕES -----
129         #atualiza valores da matriz principal
130     def atualiza_matriz(self, linha, valor):
131         self._matriz_sinais[linha] = np.roll(self._matriz_sinais[linha],
132         1, axis=0)
133         self._matriz_sinais[linha][0] = valor
134
135     def top_block(self):
136
137         if OMCT: #Se OMCT pula as derivadas e coloca as senoides direto
138             input_to_slop_channel = \
139             np.array(self._matriz_sinais["u"][0][:3])
140
141         else:
142             #deriva os sinais para posição para aceleração
143             derivative_top_input = func.deriva(
144             self._matriz_sinais["u"], self._Ts)
145
146             self.atualiza_matriz("u_derivado", derivative_top_input)
147
148             derivative_derivative_top_input = func.deriva(
149             self._matriz_sinais["u_derivado"], self._Ts)
150
151             input_to_slop_channel = (
152             derivative_derivative_top_input[:3] * \
153             self._gain_translational_acceleration)
154
155         #Segue os cálculos dos canais de translação
156         r_matrix_R_dot_input_to_slop_channel_result = np.dot(
157         self._r_matrix_R, input_to_slop_channel)
158
159         input_to_translational_channel = np.subtract(
160         r_matrix_R_dot_input_to_slop_channel_result, self._g,)
161
162         translational_channel_output = \
163         self._translational_channel_operations(
164         input_to_translational_channel)
165
166         slop_channel_output = self._slop_channel_operations(
167         input_to_slop_channel)
168

```

```

169         return translational_channel_output, slop_channel_output
170
171     def bottom_block(self, slop_channel_output):
172
173         if OMCT: #Se OMC pula as derivadas e coloca as senoides direto
174             derivative_bottom_input_with_gain = np.array(
175                 self._matriz_sinais["u"][0][3:])
176
177         else:
178             derivative_bottom_input = \
179                 self._matriz_sinais["u_derivado"][0][3:]
180             derivative_bottom_input_with_gain = (
181                 derivative_bottom_input * self._gain_angular_velocity)
182
183             #Segue os cálculos dos canais de rotação
184             derivative_bottom_input_with_gain_dot_rw_matrix_R = np.dot(
185                 derivative_bottom_input_with_gain, self._rw_matrix_R)
186
187             rotation_channel_output = self._rotation_channel_operations(
188                 derivative_bottom_input_with_gain_dot_rw_matrix_R)
189
190             bottom_block_result = np.add(
191                 rotation_channel_output, slop_channel_output + [0])
192
193             self._matrix_R1_operations(bottom_block_result)
194
195             return bottom_block_result
196
197         # -----REALIZA CÁLCULO DAS FUNÇÕES DE TRNASFERÊNCIAS --
198
199     def _translational_channel_operations(self, u_translacao) -> np.ndarray:
200
201         # Salva valores de u encontrado na matriz de dados
202         self.atualiza_matriz("u_translacaoX", u_translacao[0])
203         self.atualiza_matriz("u_translacaoY", u_translacao[1])
204         self.atualiza_matriz("u_translacaoZ", u_translacao[2])
205
206         valor_translacaoX = func.resultado_FZ(
207             self._FZ_translacaoXY,
208             self._matriz_sinais["u_translacaoX"],
209             self._matriz_sinais["y_translacaoX"],)
210
211         valor_translacaoY = func.resultado_FZ(
212             self._FZ_translacaoXY,
213             self._matriz_sinais["u_translacaoY"],
214             self._matriz_sinais["y_translacaoY"],
215         )
216
217         valor_translacaoZ = func.resultado_FZ(

```

```

217         self._FZ_translacaoZ,
218         self._matriz_sinais["u_translacaoZ"],
219         self._matriz_sinais["y_translacaoZ"],
220     ) #
221
222     # Salva valores de y encontrado na matriz de dados
223     self.atualiza_matriz("y_translacaoX", valor_translacaoX)
224     self.atualiza_matriz("y_translacaoY", valor_translacaoY)
225     self.atualiza_matriz("y_translacaoZ", valor_translacaoZ)
226
227     output = [
228     valor_translacaoX, valor_translacaoY, valor_translacaoZ + self.Z_center]
229     return output
230
231
232     def _slop_channel_operations(self, u_inclinacao: np.ndarray) -> \
233     np.ndarray:
234         g = 9.81
235         # Salva valores de u encontrado na matriz de dados
236         self.atualiza_matriz("u_inclinacaoX", u_inclinacao[0])
237         self.atualiza_matriz("u_inclinacaoY", u_inclinacao[1])
238
239         valor_inclinacaoX = func.resultado_FZ(
240         self._FZ_inclinacaoX,
241         self._matriz_sinais["u_inclinacaoX"],
242         self._matriz_sinais["y_inclinacaoX"],
243     )
244         valor_inclinacaoY = func.resultado_FZ(
245         self._FZ_inclinacaoY,
246         self._matriz_sinais["u_inclinacaoY"],
247         self._matriz_sinais["y_inclinacaoY"],
248     )
249
250     # Salva valores de y encontrado na matriz de dados
251     self.atualiza_matriz("y_inclinacaoX", valor_inclinacaoX)
252     self.atualiza_matriz("y_inclinacaoY", valor_inclinacaoY)
253
254
255     # Limita para não deixar o asin extrapolar [-g g]
256     valor_inclinacaoX = max(-g, min(g, valor_inclinacaoX))
257     valor_inclinacaoY = max(-g, min(g, valor_inclinacaoY))
258
259     tiltY1 = math.asin(valor_inclinacaoX / g)
260     tiltX1 = math.asin(-valor_inclinacaoY / g)
261
262     tiltX = func.limite_rater(
263     tiltX1, self._matriz_sinais["tiltX"][0], self._taxa*self._Ts)
264     tiltY = func.limite_rater(

```

```

265         tiltY1, self._matriz_sinais["tiltY"][0], self._taxa*self._Ts)
266
267         # salvando os novos valores na matriz de sinais
268         self.atualiza_matriz("tiltX", tiltX)
269         self.atualiza_matriz("tiltY", tiltY)
270
271         output = [-tiltX, tiltY]
272         return output
273
274     def _rotation_channel_operations(self, u_rotacao: np.ndarray) -> \
275     np.ndarray:
276         # Salva valores de u encontrado na matriz de dados
277         self.atualiza_matriz("u_rotacaoX", u_rotacao[0])
278         self.atualiza_matriz("u_rotacaoY", u_rotacao[1])
279         self.atualiza_matriz("u_rotacaoZ", u_rotacao[2])
280
281         valor_rotacaoX = func.resultado_FZ(
282             self._FZ_rotacaoXY,
283             self._matriz_sinais["u_rotacaoX"],
284             self._matriz_sinais["y_rotacaoX"],
285         )
286
287         valor_rotacaoY = func.resultado_FZ(
288             self._FZ_rotacaoXY,
289             self._matriz_sinais["u_rotacaoY"],
290             self._matriz_sinais["y_rotacaoY"],
291         )
292
293         valor_rotacaoZ = func.resultado_FZ(
294             self._FZ_rotacaoZ,
295             self._matriz_sinais["u_rotacaoZ"],
296             self._matriz_sinais["y_rotacaoZ"],
297         )
298
299         # Salva valores de y encontrado na matriz de dados
300         self.atualiza_matriz("y_rotacaoX", valor_rotacaoX)
301         self.atualiza_matriz("y_rotacaoY", valor_rotacaoY)
302         self.atualiza_matriz("y_rotacaoZ", valor_rotacaoZ)
303
304         output = [valor_rotacaoX, valor_rotacaoY, valor_rotacaoZ]
305         return output
306
307     #Calcula as matrizes de rotação
308     def _matrix_R1_operations(self, angulos: np.ndarray) -> None:
309
310         roll = angulos[0]
311         pitch = angulos[1]
312         yaw = angulos[2]
313
314         Rx = [

```

```

313         [1, 0, 0],
314         [0, math.cos(roll), -(math.sin(roll))],
315         [0, math.sin(roll), math.cos(roll)],
316     ]
317     Ry = [
318         [math.cos(pitch), 0, math.sin(pitch)],
319         [0, 1, 0],
320         [-(math.sin(pitch)), 0, math.cos(pitch)],
321     ]
322     Rz = [
323         [math.cos(yaw), -(math.sin(yaw)), 0],
324         [math.sin(yaw), math.cos(yaw), 0],
325         [0, 0, 1],
326     ]
327
328     R = np.dot(np.dot(Rz, Ry), Rx)
329
330     Rw = [
331         [math.cos(yaw) * math.cos(pitch), -(math.sin(yaw)), 0],
332         [math.cos(pitch) * math.sin(yaw), math.cos(yaw), 0],
333         [-(math.sin(yaw)), 0, 1],
334     ]
335
336     self._r_matrix_R = R
337     self._rw_matrix_R = Rw
338
339
340 def main():
341     rclpy.init()
342
343     control_system_node = ControlSystemNode()
344
345     rclpy.spin(control_system_node)
346
347     control_system_node.destroy_node()
348     rclpy.shutdown()
349
350 if __name__ == '__main__':
351     main()

```