

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE COMPUTAÇÃO
CURSO DE ENGENHARIA DE COMPUTAÇÃO

NATHÁLIA DOS SANTOS ARAÚJO

**COMPARAÇÃO ENTRE CONTROLADORES PID COM AJUSTE
DE GANHOS COM ALGORITMO GENÉTICO PARA UM VEÍCULO
AÉREO NÃO TRIPULADO DE ASA FIXA**

TRABALHO DE CONCLUSÃO DE CURSO

CORNÉLIO PROCÓPIO
2020

NATHÁLIA DOS SANTOS ARAÚJO

**COMPARAÇÃO ENTRE CONTROLADORES PID COM AJUSTE
DE GANHOS COM ALGORITMO GENÉTICO PARA UM VEÍCULO
AÉREO NÃO TRIPULADO DE ASA FIXA**

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da Universidade Tecnológica Federal do Paraná, como requisito parcial para a obtenção do título de Bacharel em Engenharia de Computação.

Orientadora: Natássya Barlate Floro da Silva
Universidade Tecnológica Federal do Paraná

CORNÉLIO PROCÓPIO
2020



Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Cornélio Procópio
Nome da Diretoria
Nome da Coordenação
Nome do Curso



TERMO DE APROVAÇÃO

Comparação entre controladores PID com ajuste de ganhos com Algoritmo Genético para um Veículo Aéreo Não Tripulado de asa fixa

por

Nathália dos Santos Araújo

Este Trabalho de Conclusão de Curso de graduação foi julgado adequado para obtenção do Título de "Bacharel em Engenharia da Computação" e aprovado em sua forma final pelo Programa de Graduação em Engenharia da Computação da Universidade Tecnológica Federal do Paraná.

Cornélio Procópio, 26/08/2020

Profa. Dra. Natássya Barlate Floro da Silva

Prof. Dr. Antônio Carlos Fernandes da Silva

Prof. Dr. Rogério Santos Pozza

"A Folha de Aprovação assinada encontra-se na Coordenação do Curso"

Dedico este trabalho a todos que me acompanharam, participaram e torceram por mim nesse trajeto. Aos que tornaram isso possível, obrigada: nós conseguimos!

AGRADECIMENTOS

Primeiramente, agradeço aos meus pais, Angela e Dagmar, pelo apoio, compreensão e carinho durante esses cinco anos. Agradeço imensamente à minha família, em especial a Marivone, Wagner, Letícia, Patrícia, Bruno e Enzo. O suporte de vocês foi essencial para que concluísse essa etapa, sou imensamente grata por ter vocês ao meu lado, independente da distância.

Agradeço à UTFPR por me acolher e conceder muitas oportunidades além da graduação. Por apoiar e dar suporte a tantos projetos, construir um ambiente acadêmico onde os corpo discente tem força. Por proporcionar tantas experiências boas, que levarei comigo para sempre, destacando o Diretório Acadêmico de Computação e o Grupo de Pesquisa EDITEC. Às pessoas que conheci durante essa jornada: muito obrigada por tudo.

Agradeço especialmente minha orientadora, Natássya, pelo trabalho aqui desenvolvido. Obrigada por sua dedicação, preocupação e disponibilidade sempre. Seu profissionalismo e comprometimento com as atividades acadêmicas é inspirador e minha admiração só cresceu ao compartilharmos este projeto.

Por fim, sou grata a todos os meus amigos, docentes e servidores que me ajudaram e apoiaram durante a graduação. Em especial, ao Bruno Braziel, Wesley Janoni, Jhonatan Canedo, Marilu Martens, Milena Barbosa e Cíntia Paião. Obrigada por toda inspiração e crescimento. Nada disso teria acontecido sem vocês.

Que nada nos defina, que nada nos sujeite. Que a liberdade seja a nossa própria substância, já que viver é ser livre. (Simone de Beauvoir)

RESUMO

ARAÚJO, Nathália. Comparação entre controladores PID com ajuste de ganhos com Algoritmo Genético para um Veículo Aéreo Não Tripulado de asa fixa. 2020. 55 f. Trabalho de Conclusão de Curso – Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2020.

Os Veículos Aéreos Não Tripulados (VANTs) são pequenas aeronaves que podem ser controladas de forma remota ou configuradas para voar de forma autônoma, sem a atuação direta de um ser humano. Dada a popularização crescente desses veículos, assim como o aumento de sua utilização em tarefas civis, novos modelos e novas técnicas de controle têm surgido, visando aumentar não só o custo-benefício, mas também o desempenho operacional dos mesmos. Nesse contexto, este trabalho tem como foco VANTs de asa fixa – modelo de alta eficiência e baixo custo em geral –, cuja autonomia de voo é alcançada por estratégias de controle executadas por meio do Piloto Automático da aeronave. Portanto, têm-se como objetivo a comparação entre duas dessas estratégias: o PID para posição e os controladores PID para posição e velocidade, a fim de avaliar o desempenho do aeronave Cessna Skyhawk em ambos, visto que possuem complexidade e procedimentos distintos. O modelo de aeronave utilizado é disponibilizado pelo simulador de voo X-Plane 11 e os códigos dos controles são desenvolvidos em linguagem M, assim, os testes de *Software in the Loop* (SIL) são realizados pelo simulador integrado ao MATLAB, que executa os controladores. Para realizar a integração entre as versões atuais dos dois programas, foi necessário estabelecer a comunicação direta entre eles, por meio de um protocolo que utiliza o *User Datagram Protocol* (UDP) em linguagem M, a fim de obter os valores em tempo real dos ângulos, posição e velocidade da aeronave. A partir desses valores, as estratégias de controle podem realizar o cálculo da atuação necessária para que o VANT tenha um voo estável. Os valores obtidos por cada controlador são atribuídos aos atuadores do veículo, responsáveis por seu controle físico na simulação. Com a troca de dados entre os programas, o fluxo de leitura e escrita das variáveis de controle podem ser realizados durante o tempo real de simulação.

Palavras-chave: Sistemas de Controle, *Software in the Loop*, Veículo Aéreo Não Tripulado

ABSTRACT

ARAÚJO, Nathália. Comparison between PID controllers and gains adjustment with Genetic Algorithm for a fixed wing Unmanned Aerial Vehicle. 2020. 55 f. Trabalho de Conclusão de Curso – Curso de Engenharia de Computação, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2020.

Unmanned Aerial Vehicles (UAVs) are small aircrafts capable of being controlled remotely or operate autonomously, without a human aboard. Due to technological advancements and its increasing use in civilian tasks, new models and new control techniques to increase the cost-benefit and its performance are emerging. In this context, this work is focused on fixed wing UAV – a low cost and high performance model –, which flight autonomy is achieved with control strategies implemented in its autopilot unit. In this scenario, this work aims to compare two control strategies: PID for position and PID for position and velocity, to evaluate Cessna Skyhawk's performance on both of them since their complexity are different. This aircraft is available on X-Plane 11 and the control codes are implemented on M. Thus, the Software in the Loop (SIL) experiments are executed with the flight simulator combined with MATLAB. To accomplish the communication between both softwares, it was implemented a protocol based on User Datagram Protocol (UDP) to get real time values of angles, positions and velocities from the aircraft on the simulator. With these values both PID controllers can define the required effort to the UAV to achieve a stable flight. Those values returned from the controllers are commanded to the aircraft's control surfaces that are responsible for its control during the simulation. Through communication protocol implemented on both programs, reading and writing data can be performed on simulation real time.

Keywords: Control Systems, Software in the Loop, Unmanned Aerial Vehicle

LISTA DE FIGURAS

Figura 1 – Estrutura do piloto automático.	6
Figura 2 – Sistema de coordenadas inercial NED (<i>North-East-Down</i>) e sistema de coordenadas fixado no corpo da aeronave. A atitude da aeronave está representada como rotações em cada um dos eixos do sistema de coordenadas fixado no corpo, como: <i>roll</i> ϕ , <i>pitch</i> θ e <i>yaw</i> ψ	6
Figura 3 – Aeronave do tipo asa fixa com superfícies de controle destacadas.	7
Figura 4 – Sistema de controle em malha fechada.	8
Figura 5 – Estratégia PID para posição.	9
Figura 6 – Controlador PID da intensidade da velocidade da aeronave.	10
Figura 7 – Estratégia PID para posição e velocidade.	10
Figura 8 – Fluxograma do <i>script</i> de simulação utilizado nos experimentos.	22
Figura 9 – Processo de obtenção dos ganhos utilizando o Algoritmo Genético.	24
Figura 10 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição com Algoritmo Genético, sem seguimento de caminho.	26
Figura 11 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição com Algoritmo Genético, sem seguimento de caminho.	26
Figura 12 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição com Algoritmo Genético, sem seguimento de caminho.	27
Figura 13 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição com Algoritmo Genético, com seguimento de caminho.	27
Figura 14 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição com Algoritmo Genético, com seguimento de caminho.	28
Figura 15 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição com Algoritmo Genético, com seguimento de caminho.	28
Figura 16 – Gráfico de comparação entre os conjuntos de ganhos para as variáveis <i>roll</i> e <i>pitch</i> do controlador PID para posição.	30
Figura 17 – Gráfico de comparação entre os conjuntos de ganhos para o ângulo de <i>sideslip</i> do controlador PID para posição.	30
Figura 18 – Gráfico de comparação entre os conjuntos de ganhos para a variável de velocidade do controlador PID para posição	31
Figura 19 – Comportamento de cada conjunto de ganho para o ângulo de <i>roll</i> em relação aos valores de referência na estratégia de controle para posição.	32
Figura 20 – Comportamento de cada conjunto de ganhos para o ângulo de <i>pitch</i> em relação aos valores de referência na estratégia de controle para posição.	33
Figura 21 – Comportamento de cada conjunto de ganhos para o ângulo de <i>sideslip</i> em relação aos valores de referência na estratégia de controle para posição.	33

Figura 22 – Comportamento de cada conjunto de ganhos para velocidade em relação ao valor de referência na estratégia de controle para posição.	34
Figura 23 – Comportamento de cada conjunto de ganhos durante o seguimento de caminho para a estratégia de controle para posição.	34
Figura 24 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição e velocidade com Algoritmo Genético, sem seguimento de caminho.	35
Figura 25 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição e velocidade com Algoritmo Genético, sem seguimento de caminho.	35
Figura 26 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição e velocidade com Algoritmo Genético, sem seguimento de caminho.	36
Figura 27 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição e velocidade com Algoritmo Genético, com seguimento de caminho.	36
Figura 28 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição e velocidade com Algoritmo Genético, com seguimento de caminho.	37
Figura 29 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição e velocidade com Algoritmo Genético, com seguimento de caminho.	37
Figura 30 – Gráfico de comparação entre os conjuntos de ganhos para as variáveis <i>roll</i> , <i>pitch</i> do controlador PID para posição e velocidade.	39
Figura 31 – Gráfico de comparação entre os conjuntos de ganhos para as velocidades angulares Q e P do controlador PID para posição e velocidade.	40
Figura 32 – Gráfico de comparação entre os conjuntos de ganhos para o ângulo de <i>sideslip</i> do controlador PID para posição e velocidade.	40
Figura 33 – Gráfico de comparação entre os conjuntos de ganhos para a variável de velocidade do controlador PID para posição e velocidade.	41
Figura 34 – Comportamento de cada conjunto de ganho para o ângulo de <i>roll</i> em relação aos valores de referência na estratégia de controle para posição e velocidade.	42
Figura 35 – Comportamento de cada conjunto de ganhos para o ângulo de <i>pitch</i> em relação aos valores de referência na estratégia de controle para posição e velocidade.	42
Figura 36 – Comportamento de cada conjunto de ganhos para o ângulo de <i>sideslip</i> em relação aos valores de referência na estratégia de controle para posição e velocidade.	43
Figura 37 – Comportamento de cada conjunto de ganhos para velocidade em relação ao valor de referência na estratégia de controle para posição e velocidade.	43

Figura 38 – Comportamento de cada conjunto de ganhos durante o seguimento de caminho para a estratégia de controle para posição e velocidade.	44
Figura 39 – Comportamento de cada estratégia de controle em relação ao valor de referência para o ângulo roll.	46
Figura 40 – Comportamento de cada estratégia de controle em relação ao valor de referência para o ângulo pitch.	46
Figura 41 – Comportamento de cada estratégia de controle em relação ao valor de referência para o ângulo de sideslip.	47
Figura 42 – Comportamento de cada estratégia de controle em relação ao valor de referência para a variável de velocidade.	47
Figura 43 – Comportamento de cada conjunto de ganhos durante o seguimento de caminho.	48

LISTA DE TABELAS

Tabela 1 – Ganhos utilizados para a estratégia de controle para posição.	45
Tabela 2 – Ganhos utilizados para a estratégia de controle para posição e velocidade. .	45

LISTA DE ALGORITMOS

3.1	Estrutura de criação do objeto socket.	16
3.2	Envio de uma mensagem pelo objeto socket.	16
3.3	Recebimento de um pacote por meio do objeto de comunicação socket.	16
3.4	Encerramento do objeto socket.	16
3.5	Estrutura da mensagem de requisição RREF.	17
3.6	Formatação da mensagem no MATLAB para ser enviada ao X-Plane 11.	17
3.7	Estrutura da mensagem de resposta RREF.	18
3.8	Estrutura da mensagem de atribuição de valores DREF.	18
3.9	Formatação da estrutura DREF.	18

SUMÁRIO

1 – INTRODUÇÃO	1
1.1 Objetivos	2
1.2 Organização do texto	3
2 – FUNDAMENTAÇÃO TEÓRICA	5
2.1 Aeronaves de asa fixa e Piloto Automático	5
2.2 Sistemas de Controle	7
2.2.1 Estratégias de Controle	9
2.3 Trabalhos Relacionados	11
3 – DESENVOLVIMENTO	15
3.1 Sistema de comunicação: MATLAB e X-Plane	15
3.1.1 Requisitando dados do X-Plane	16
3.1.2 Atribuindo um valor a uma variável do X-Plane	18
3.2 Seguimento de caminho	19
3.3 Testes de <i>Software In the Loop</i>	20
3.3.1 Experimentos de simulação	20
3.3.2 Otimização com o Algoritmo Genético para o ajuste de ganhos	21
4 – ANÁLISE DOS RESULTADOS	25
4.1 Ganhos obtidos para a estratégia de controle de posição	25
4.2 Ganhos obtidos para a estratégia de controle de posição e velocidade	32
4.3 Comparação entre as estratégias de controle	44
5 – CONCLUSÕES	49
5.1 Trabalhos futuros	50
Referências	53

1 INTRODUÇÃO

Os Veículos Aéreos Não Tripulados (VANTs), originalmente desenvolvidos para fins militares, atualmente também têm sido utilizados em aplicações civis e comerciais. São aeronaves que podem ser controladas remotamente ou de forma autônoma, com sistemas de controle que as mantêm estáveis durante o voo (VALAVANIS, 2007). Os avanços tecnológicos nesta área permitiram a criação de modelos menores, que são os modelos mais populares e de uso corrente. Embora menos elaborados, estes podem exercer as mesmas funções que as grandes aeronaves, com custo reduzido e múltiplos voos, sendo igualmente eficazes. As aplicações dos VANTs são extremamente amplas e significativas, como monitoramento ambiental (detecção de poluição e queimadas), mapeamento de terrenos, vigilância aérea, assistências variadas e afins (AUSTIN, 2010; ZHANG; KOVACS, 2012; BEARD; MCLAIN, 2012).

O nome VANT não se refere apenas à aeronave em si, mas a todo o sistema empregado, incluindo os microcontroladores e microprocessadores, sensores, software, interfaces e o hardware como um todo (BEARD; MCLAIN, 2012). Por não possuir interferência humana em sua atuação, um veículo aéreo autônomo necessita de um sistema de controle para guiar seu voo e a atitude adequada da aeronave para a aplicação designada (atitude é a posição angular do veículo, definida pelos ângulos *roll*, *pitch* e *yaw*). Um sistema, de maneira geral, é um conjunto de componentes que, unidos, trabalham para atingir um objetivo. Sendo assim, um sistema de controle é um modelo matemático que gerencia os dados de entrada e saída de um conjunto para que esse cumpra o seu propósito (OGATA, 2010).

Este trabalho tem como cerne o controle autônomo para aeronaves de asa fixa, cuja dinâmica de voo é baseada em três ângulos, que representam a rotação do veículo em três dimensões sobre seu centro de massa: *roll*, *pitch* e *yaw*. Para isso, foram utilizados os programas MATLAB e X-Plane, sendo o primeiro responsável pelos controladores, e o segundo pela simulação de voo para a aeronave Cessna Skyhawk. Embora existam outras plataformas que suportem este tipo de operação, o X-Plane é utilizado por fornecer a maior fidelidade física e de visualização 3D (CRAIGHEAD et al., 2007), além de se adequar aos modelos de aeronave e de voo e suportar a execução de *Software in the Loop* (SIL), que são os testes de voo realizados por meio de simulação e ferramentas em um ambiente completamente virtual. Esta técnica permite validar estruturas do piloto automático sem o risco de perder a aeronave física em diferentes condições de voo, visto que na simulação há a presença de distúrbios e possíveis adversidades, para que o teste se assemelhe o máximo possível às condições reais (VALAVANIS; VACHTSEVANOS, 2015).

Dessa forma, pôde-se validar e avaliar o desempenho de duas estruturas baseadas no controlador Proporcional Integral e Derivativo (PID) para o piloto automático da aeronave Cessna Skyhawk (modelo de asa fixa fornecido pelo próprio simulador). Controladores do tipo PID são utilizados em sistemas de malha fechada, como o da aeronave, onde os valores de

saída do controlador são considerados para o próximo cálculo de controle. A primeira estrutura é nomeada como PID para posição e a segunda é chamada de PID para posição e velocidade e embora sejam baseadas no mesmo tipo de controlador, possuem complexidades diferentes. A principal diferença entre as estratégias se dá pelo fato de que o PID para posição possui operação mais simples, controlando apenas as posições angulares da aeronave, enquanto o PID para posição e velocidade, que pode ser referenciado como PID em cascata (pois utiliza controladores interligados) controla as posições e a velocidade, e pode fornecer uma execução mais precisa das manobras de voo.

Os controladores recebem valores iniciais de referência para os ângulos e, a partir destes, é calculada a atuação necessária para que a aeronave se estabilize. O cálculo é realizado pela diferença entre os ângulos referenciais e os ângulos obtidos durante o voo e os valores são enviados para o sistema de controle do veículo. Para obter esses valores e enviar os novos em tempo real de simulação, foi necessária a integração entre o MATLAB e o X-Plane 11. A comunicação entre os programas não é nativa nas versões atuais, portanto, foi implementada a troca de informações por meio de um protocolo específico do simulador, que utiliza o protocolo *User Datagram Protocol* (UDP). Os comandos partem do MATLAB e são enviados para a porta 49000 do X-Plane, que retorna dados ou executa ações dentro do programa de acordo com o que foi recebido.

1.1 Objetivos

Este trabalho tem como objetivo principal a comparação entre duas estratégias de controle baseadas no controlador PID para VANTs com asa fixa, por meio de testes de *Software in the Loop*. A primeira estratégia, chamada de PID para posição, utiliza controladores PID que possuem como entrada os ângulos de atitude enquanto que a segunda estratégia, chamada de PID para posição e velocidade, utiliza controladores PID em cascata que recebem como entrada os ângulos de atitude e as velocidades angulares de *roll* e *pitch*. Para a execução dos testes, utiliza-se o simulador de voos X-Plane e, para a execução dos controladores, o *software* MATLAB. Para que os programas atuem em conjunto nas versões correntes, é necessário implementar um protocolo de comunicação direta entre eles, sendo essa parte fundamental para alcançar o objetivo inicial. Outra parte de suma importância para a conclusão deste projeto trata-se do ajuste de ganhos dos controladores, pois, para que o PID funcione de maneira correta é necessário ajustar cada um de seus ganhos (proporcional, integral e derivativo). Este ajuste pode ser feito por testes empíricos, entretanto, melhores resultados são obtidos por meio de algoritmos de otimização. Sendo assim, busca-se encontrar os valores ideais de cada ganho por meio de Algoritmos Genéticos.

1.2 Organização do texto

Este primeiro capítulo apresenta, de maneira sucinta, o que é um VANT e quais são seus usos, assim como uma definição básica sobre sistemas de controle e uma breve descrição do projeto e tecnologias utilizadas. O Capítulo 2 traz o embasamento teórico necessário para compreender as técnicas utilizadas neste trabalho, aprofundando-se nos conceitos e terminologias empregados. No Capítulo 3 é detalhado como foi desenvolvida a comunicação entre o simulador X-Plane e o MATLAB, e também implementação das estruturas para as simulações de SIL e uso dos controladores PID. O Capítulo 4 mostra os resultados obtidos a partir das simulações de cada estratégia de controle adotada por meio de gráficos, a análise de cada uma e a comparação entre elas. Por fim, o Capítulo 5 trata das conclusões obtidas por meio deste trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os conceitos fundamentais para o entendimento do projeto.

2.1 Aeronaves de asa fixa e Piloto Automático

As características de um VANT variam de acordo com seu uso, necessidade e particularidades. Embora existam inúmeros modelos, neste projeto destaca-se o modelo de asa fixa. Como o próprio nome sugere, esses VANTs possuem asas fixas em seu corpo, similares a um avião tradicional. Isso implica em grande autonomia, maior sustentação e baixo gasto energético (AUSTIN, 2010).

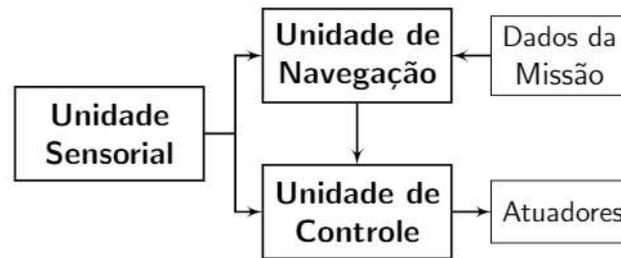
Para operar um veículo de maneira independente, isto é, sem interferência interna ou externa de controle, como um piloto ou controle remoto, é utilizado o Piloto Automático. Esta unidade garante a autonomia da aeronave, pois as estruturas de controle lá aplicadas são capazes de manter sua estabilidade durante a missão. Visto que os veículos de asa fixa possuem operação complexa com múltiplos estágios de voo, que diferem entre si, é necessário que o VANT tenha um piloto automático íntegro para ser utilizado de forma autônoma (SILVA, 2018).

O Piloto Automático garante a autonomia da aeronave sem que seu desempenho seja afetado, possibilitando o enfoque na atividade para qual o voo foi designado. Através deste, é mantida a estabilidade durante todas as manobras necessárias para a missão e o operador do VANT pode focar nos requisitos da própria aplicação, como em dispositivos de monitoramento, câmeras, entre outros. Seu núcleo é dividido em unidades com responsabilidades individuais, para controlar os atuadores da aeronave e para que seja possível executar as demais tarefas, a fim de estabilizar os componentes periféricos e também o veículo (VALAVANIS; VACHTSEVANOS, 2015).

O Piloto Automático é dividido de acordo com suas principais funções, como mostra a Figura 1. A Unidade Sensorial é responsável por fornecer às outras unidades informações sobre o estado da aeronave, como sua posição espacial, velocidade e os ângulos de atitude (*roll*, *pitch* e *yaw*), obtidas por meio dos sensores. A Unidade de Navegação gerencia os estágios do voo e fornece, para a Unidade de Controle, um estado de referência que é composto pela magnitude da velocidade e pela atitude. Ainda, assegura que a trajetória estabelecida seja seguida por meio dos algoritmos de seguimento de trajetória durante o voo. E, por fim, a Unidade de Controle é responsável por manter a estabilidade da aeronave enviando comandos para os atuadores. Ao receber o estado de referência da Unidade de Navegação, calcula os comandos com base em diferentes modelos de controle para cada estágio de voo (SILVA et al., 2018).

As estratégias de controle são alocadas na Unidade de Controle, que recebe os dados

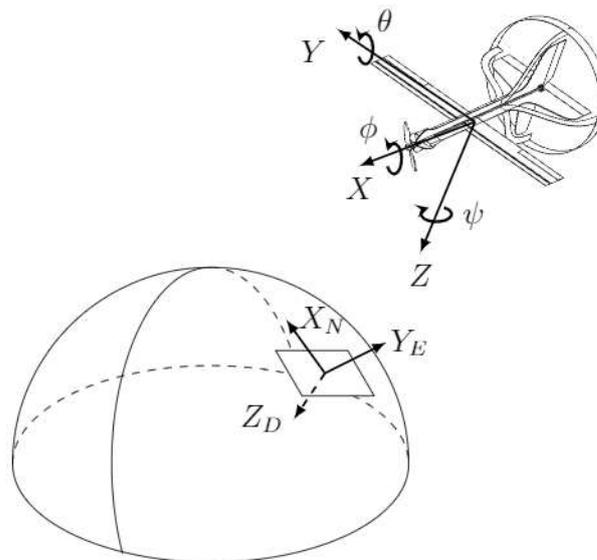
Figura 1 – Estrutura do piloto automático.



Fonte: Silva et al. (2018).

de entrada das outras unidades e, assim, pode calcular a atuação necessária da aeronave para que os valores de referência fornecidos pela Unidade de Navegação sejam seguidos. Os atuadores, por sua vez, trabalham para estabilizar a atitude do veículo, dada pelos ângulos de Tait-Bryan: *roll* ϕ , *pitch* θ e *yaw* ψ , como mostra a Figura 2. O ângulo ϕ representa o balanço (eixo longitudinal), já o ângulo θ expressa o eixo lateral, dado pelo empinamento da aeronave e o ψ representa a ação no eixo vertical, conhecido como guinada. A letra (β) representa o ângulo de *sideslip* da aeronave, conhecido em português como ângulo de derrapagem. Este, representa a estabilidade lateral da aeronave.

Figura 2 – Sistema de coordenadas inercial NED (*North-East-Down*) e sistema de coordenadas fixado no corpo da aeronave. A atitude da aeronave está representada como rotações em cada um dos eixos do sistema de coordenadas fixado no corpo, como: *roll* ϕ , *pitch* θ e *yaw* ψ .

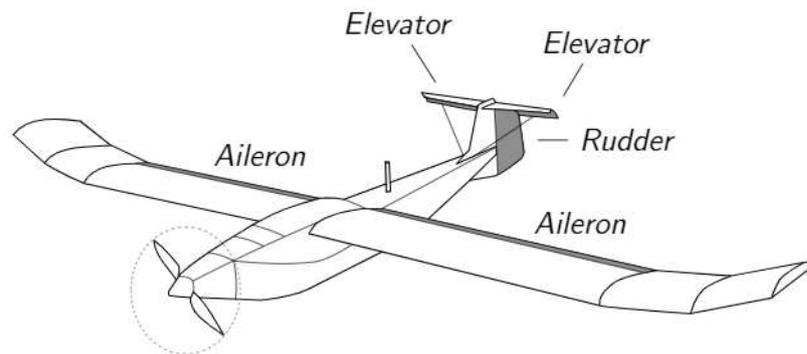


Fonte: Silva et al. (2018).

Portanto, a Unidade de Controle trabalha com estratégias para estabilizar a aeronave durante o voo. Os valores de entrada são as referências para os ângulos da atitude e o valor

em tempo real dos mesmos. Como saída, a unidade provê os comandos para os atuadores que controlam o veículo. Cada atuador é responsável por controlar um dos ângulos que compõe a atitude da aeronave. Para os modelos de asa fixa, a superfície de controle *aileron* controla o ângulo *roll*, o *elevator* controla o *pitch*, o *rudder* é responsável pelo ângulo de *sideslip* e, por fim, o atuador *throttle* regula da velocidade do veículo. A Figura 3 apresenta as superfícies de controle de uma aeronave asa fixa.

Figura 3 – Aeronave do tipo asa fixa com superfícies de controle destacadas.



Fonte: Silva et al. (2018).

2.2 Sistemas de Controle

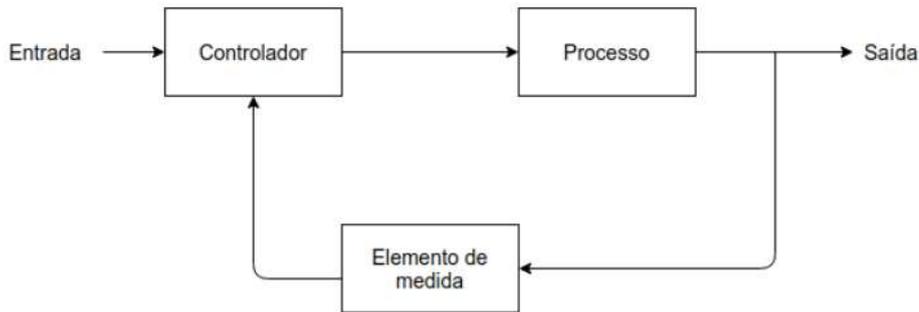
Por definição, um sistema é a combinação de componentes que atuam, em conjunto, para concretizar um objetivo. De maneira análoga, um sistema de controle é um modelo matemático responsável por comandar um conjunto com base em suas entradas e saídas. Esse controlador pode ser do tipo malha aberta ou fechada. Para que um VANT opere de forma autônoma é utilizado o controle em malha fechada, onde o sinal de saída do sistema (da aeronave, no caso) tem efeito direto na ação do controlador, definindo-o como um sistema realimentado. O sinal de saída é medido e se não for satisfatório, é utilizado para o cálculo da nova atuação do controlador. Isso se repete até que seja obtida a saída esperada, como exemplificado na Figura 4 (OGATA, 2010).

O sistema de controle deve proporcionar um comportamento estável e ser capaz de reduzir ou minimizar erros, assim, é construído um modelo específico para cada caso, sendo considerados o objetivo, o modelo de projeto, as especificações dos componentes e o índice de desempenho esperado, respeitando os parâmetros definidos (OGATA, 2010).

As estratégias implementadas na Unidade de Controle da aeronave utilizam controladores do tipo Proporcional Integral Derivativo (PID), onde são utilizados os erros atuais, posteriores e previstos para calcular a atuação da aeronave.

Um controlador PID é responsável por realizar o ajuste fino em sistemas de malha fechada, onde o ajuste de seus ganhos resultam em um sinal de controle enviado para o processo, cuja saída é lida pela malha de realimentação e, assim, retornada para as três partes que o

Figura 4 – Sistema de controle em malha fechada.



Fonte: (OGATA, 2010).

compõem. A ação Proporcional (P) é responsável por minimizar o sinal de erro; a ação Integral (I) alimenta a integral do erro, a fim de zerar o componente de erro estacionário; e, por fim, a ação Derivativa (D) atua sob a taxa de variação do erro para obtê-lo antecipadamente. A combinação PID visa a redução do erro estacionário e estabilidade do sistema, calculando a resposta de saída do atuador (NISE, 2012).

Os controladores PID que são utilizados nesse trabalho são implementados como um controle discreto calculado por (1), com os seguintes termos: proporcional descrito por (2), integral com *anti-windup* obtido por *backward difference* resultando em (3) e derivativo com filtro de primeira ordem e também obtido por *backward difference* resultando em (4). O termo t_k representa os instantes em que os valores da entrada são recebidos; $y_{ref}(t_k)$ é o valor de referência da entrada no instante t_k ; $y(t_k)$ é o valor atual da entrada no instante t_k ; K_P , K_I e K_D são os ganhos do controlador; Δt é o período de discretização, considerado o tempo com que os valores são recebidos, e T_f é o parâmetro do filtro do termo do derivativo (ÅSTRÖM; HÄGGLUND, 2006).

$$u(t_k) = P(t_k) + I(t_k) + D(t_k) \quad (1)$$

$$P(t_k) = K_P(y_{ref}(t_k) - y(t_k)) \quad (2)$$

$$I(t_k) = I(t_{k-1}) + K_I \Delta t (y_{ref}(t_k) - y(t_k)) \quad (3)$$

$$D(t_k) = \frac{T_f}{T_f + \Delta t} D(t_{k-1}) + \frac{K_D}{T_f + \Delta t} (y(t_k) - y(t_{k-1})) \quad (4)$$

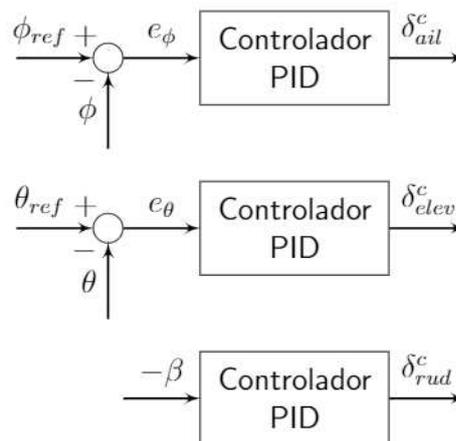
A diferença entre os valores de entrada dos controladores precisa ser realizada de uma forma diferenciada quando esses valores são ângulos devido às descontinuidades que acontecem

em múltiplos de 2π . Portanto, quando as entradas são ângulos da atitude, a diferença entre os valores de referência e os valores recebidos foram calculados pela função *angdiff* do MATLAB, retornando valores entre $[-\pi, \pi]$.

2.2.1 Estratégias de Controle

Ambas as estratégias de controle utilizadas para o desenvolvimento deste projeto são formadas por controladores PID, conforme a descrição dada na Seção 2.2. A primeira, chamada de PID para posição trabalha com um modelo simplificado da Unidade de Controle da aeronave. Os controladores são independentes e calculam separadamente a ação para cada atuador, que é responsável por estabilizar apenas um dos ângulos da atitude do veículo, como mostra a Figura 5.

Figura 5 – Estratégia PID para posição.

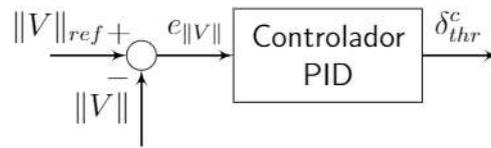


Fonte: Produzida pela autora.

Os valores de entrada são os erros referentes aos ângulos *roll*, e_ϕ , e *pitch*, e_θ . Para o ângulo de *sideslip* β é utilizado apenas o valor medido, pois é esperado que a referência seja sempre igual ou próximo a 0. Para os outros, o erro de cada um deles é calculado durante o teste SIL pela diferença entre o valor desejado para cada e o valor obtido através do X-Plane em tempo real de simulação. Então, a atuação necessária para estabilizar a aeronave é calculada pelo PID. Os valores de saída são as deflexões de cada superfície de controle, δ_{ail}^c para *aileron*, δ_{elev}^c para *elevator* e δ_{rud}^c para *rudder*, que são enviados ao simulador.

Para o controle da velocidade da aeronave, a fim de que seja sempre constante, é utilizado o controlador descrito na Figura 6, que recebe como entrada o erro da velocidade $e_{\|V\|}$, obtido de forma análoga aos erros dos ângulos de atuação. A partir do cálculo dos erros de intensidade é gerado o valor que é passado para o acelerador do motor, δ_{thr}^c , controlado pelo *throttle* na simulação. O controlador apresentado se aplica às duas estratégias de controle deste trabalho.

Figura 6 – Controlador PID da intensidade da velocidade da aeronave.



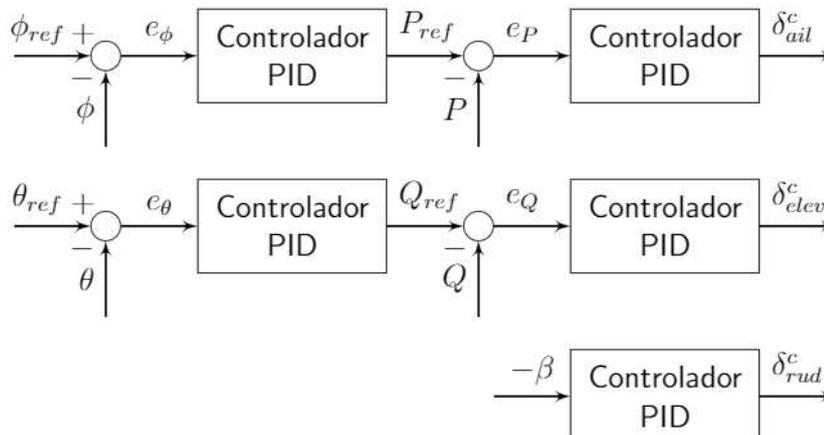
Fonte: Produzida pela autora.

A segunda estratégia, nomeada como PID para posição e velocidade, possui uma estrutura de controle em cascata, com uma abordagem mais complexa. Tem como método de operação controladores PID interligados, de forma que a saída de um torna-se parâmetro para o cálculo do valor de entrada do outro, como pode ser observado na Figura 7. Essa estratégia não considera apenas os ângulos de atitude da aeronave, mas também a velocidade angular referente a cada um destes.

Os valores de atuação para o ângulo de *sideslip* são calculados, novamente, de maneira simples para que seu valor seja o mesmo da referência. Já os ângulos *roll* e *pitch* tem o cálculo do erro realizado da mesma forma que o PID para posição, e esse erro é utilizado como parâmetro de entrada para o primeiro controlador. Após os cálculos, este controlador tem como saída o valor de referência para a velocidade de cada ângulo e, assim, essa velocidade obtida pelo primeiro PID serve de entrada para o segundo.

O segundo controlador compara o valor de referência para a velocidade angular com o seu valor real, obtido durante o *loop* de simulação. A diferença entre esses dois valores é o erro da variável, utilizado para calcular a saída final da estratégia. As variáveis P e Q referem-se às velocidades angulares de *roll* e *pitch*, respectivamente. A saída dos controladores PID em cascata são os valores de deflexão para as superfícies de controle, que são passados aos atuadores da aeronave para que o voo seja estável e seja possível seguir uma trajetória predefinida.

Figura 7 – Estratégia PID para posição e velocidade.



Fonte: Produzida pela autora.

Para que ambas as estratégias atuem de maneira satisfatória, é necessário ajustar os ganhos de cada termo, K_P , K_I e K_D que, respectivamente, representam a parte Proporcional, Integral e Derivativa de cada controlador utilizado. Neste trabalho serão analisados três conjuntos de ganhos para cada estratégia. O primeiro conjunto é obtido de maneira manual, por tentativa e erro; os outros conjuntos de ganhos serão ajustados automaticamente, utilizando uma otimização por Algoritmo Genético (AG) (MITCHELL, 1998) para encontrar valores adequados a cada variável.

2.3 Trabalhos Relacionados

Para o entendimento, motivação e desenvolvimento deste trabalho foram considerados oito artigos que tratam dos temas abordados e aplicações aqui descritas. Os trabalhos relacionados remetem ao desenvolvimento de VANTs, à utilização de *Software* e *Hardware in the Loop* (SIL e HIL, respectivamente) para testes, o uso do simulador de voos X-Plane 10 e, também, ao desenvolvimento de controles para aeronaves específicas, que segue uma abordagem semelhante à apresentada aqui.

Para embasar a utilização e criação de pequenas aeronaves autônomas, Johnson e Fontaine (2001) descrevem como reduzir os custos do desenvolvimento de um VANT sem que a aeronave tenha perdas, por meio da seleção de materiais adequados, otimização de hardware e componentes, atuação correta dos sensores e design do produto. Atuando com o conceito de baixo custo, fomentam o uso de ferramentas de simulação de voo, como SIL, para que não exista risco à estrutura física e de *hardware* da aeronave, reduzindo também a quantidade de testes para voos complexos.

Como uma referência mais recente sobre a utilização do MATLAB em conjunto ao X-Plane, há o trabalho de Kimathi (2017) que se assemelha a este não só pela escolha dos programas, mas também por tratar de VANTs de asa fixa. Nesse projeto, é utilizada a décima versão do simulador de voo para realizar testes de SIL, a fim de comprovar a performance de um controlador que utiliza o modelo de Aprendizagem por Reforço (mais conhecido pelo termo em inglês, *Reinforcement Learning* ou RL), onde busca-se que a aeronave aprenda a mapear situações e agir de acordo com o que lhe é designado. Destaca-se que a performance do controlador é mostrada usando simulações em tempo real, onde o MATLAB e X-Plane 10 são integrados para que a comunicação bidirecional seja feita em tempo corrente, evidenciando a eficiência desses programas em conjunto.

Ainda tratando-se da simulação de aeronaves, Michailidis et al. (2019) trazem o teste e implementação do *Software in the Loop* para um VANT de asa fixa. Mostram a aplicação e configuração do ambiente de simulação utilizando os programas MATLAB e X-Plane, atribuindo os dados gerados pelo simulador às entradas do controlador e coletando a resposta da aeronave. Analogamente, o trabalho proposto nessa monografia utiliza as mesmas ferramentas e forma de simulação, porém propõe a utilização de uma nova comunicação entre os programas utilizados, que se encontram em versões mais recentes.

Até a décima versão do X-Plane, sua comunicação com o MATLAB se dá por meio de um *plugin* externo, descrito e desenvolvido por [Ito et al. \(2017\)](#). Inicialmente, tanto o desenvolvimento quanto o teste de controle para uma aeronave acontecia no MATLAB e em linguagem M, mas impossibilitava a simulação simultânea de múltiplas aeronaves (para formar um esquadrão) e a comunicação entre elas. Então, desenvolveu-se um *plugin*, uma extensão do simulador, onde o código inicial é convertido para a linguagem C/C++ para que seja interpretado. A memória das operações são compartilhadas e, assim, podem operar em conjunto. Essa interação provê um ambiente mais flexível, que auxilia na implementação de novas tecnologias e facilita quaisquer alterações que necessitem ser feitas no controlador. Dessa forma, possibilitou a simulação simultânea das aeronaves com controladores distintos e o compartilhamento de recursos das aplicações.

De forma semelhante à comunicação implementada neste trabalho entre os programas citados, que permite a troca de mensagens de forma direta por meio das portas UDP do X-Plane 11 e MATLAB, no trabalho de [Bittar et al. \(2014\)](#) é proposto um guia para a aplicação de SIL para VANTs utilizando o X-Plane e o Simulink, realizando a troca de informações entre estes por meio do protocolo UDP. Pela estrutura do simulador e do Simulink, os dados são passados sequencialmente por parâmetros, de forma que ambos os programas podem compreender e enviar novas informações. Esse artigo ilustra de forma clara o funcionamento do protocolo, semelhante à proposta situada, com a diferença que o novo protocolo de comunicação do X-Plane utiliza uma nova estrutura dos dados.

A produção de [Silva et al. \(2017\)](#) engloba a maioria dos temas abordados e mostra o desenvolvimento do AVALON, um Veículo Aéreo Não Tripulado de asa fixa, com destaque ao funcionamento de sua unidade de Piloto Automático. Os resultados dos testes realizados, baseados nas equações de movimento da aeronave, provaram a eficiência do VANT e sua resposta à técnicas de controle. Embora feita a implementação de um controlador, o artigo mostra apenas o de controle PID para posição e utiliza um modelo cinemático e dinâmico da aeronave, que possui uma linearização e simplificação das forças aerodinâmicas. Em relação ao trabalho aqui desenvolvido, destaca-se que uma das estruturas de controle baseadas em PID considera não só a posição da aeronave, mas também as velocidades angulares de cada variável. Os experimentos também diferem entre si, visto que os testes deste projeto serão realizados em um simulador de voos, o X-Plane 11.

Tem-se também o trabalho de [Atlas et al. \(2015\)](#) onde é tratado o conceito de *Hardware in the Loop* (HIL) como forma de testar os algoritmos de controle para o Piloto Automático de um VANT e, também, seus sensores e superfícies de controle, diminuindo a necessidade de voos reais para teste. O projeto conta com um protótipo físico que simula o comportamento da aeronave por meio dos ângulos de atitude, onde a simulação é feita no X-Plane 10 e os valores e comandos são passados para o microcontrolador do protótipo via USB, por uma porta serial virtual. De maneira similar ao projeto aqui desenvolvido, é utilizado um controlador PID para direcionar o veículo, mas os ganhos K_P , K_I e K_D são estimados

pelo *Control and Estimation Toolbox*, enquanto aqui é utilizado o Algoritmo Genético.

Por fim, o trabalho de [Rahimi, Hajighasemi e Sanaei \(2013\)](#) traz uma comparação entre estratégias de controle para o Piloto Automático de um VANT. São comparados um controlador PID, um Regulador Quadrático Linear, uma técnica de controle baseada em Lógica Fuzzy, e um controlador PID com otimização dos ganhos Proporcional, Integral e Derivativo por meio de Lógica Fuzzy (chamado de *Self-tuning Fuzzy PID Controller*) para mostrar a eficiência e performance de um veículo com base em seus ângulos de atitude. Antes mesmo dos resultados, é destacada a vantagem do controlador PID, por ser uma estrutura clássica da literatura e relativamente simples, com fácil ajuste de ganhos. O *Self-tuning Fuzzy PID Controller* tem predileção por combinar dois métodos consolidados.

Esse controlador melhora a performance dinâmica do PID com base em algoritmos inteligentes, para afinar seus parâmetros de ganho, de forma análoga ao que é feito neste trabalho com o AG. Semelhantemente, as entradas da otimização são os sinais, os erros e a derivação dos erros obtidos em tempo de simulação, e a saída do algoritmo são os coeficientes de ganho K_P , K_I e K_D . Nos testes apresentados no trabalho referido, a estratégia de controle que apresentou melhor performance e obteve melhor resposta em relação às outras foi a que utiliza Lógica Fuzzy para o refino de ganhos do PID, sinalizando que um controlador PID combinado de um algoritmo para realizar o ajuste de ganhos pode obter um resultado mais do que satisfatório para um VANT controlado por seu Piloto Automático.

3 DESENVOLVIMENTO

Para a implementação dos sistemas de controle e para a realização da leitura e escrita dos valores de referência da aeronave é utilizado o software MATLAB (MATrix LABoratory) e sua codificação nativa, a linguagem M. Este é um programa comercial de alta performance desenvolvido pela empresa MathWorks, voltado para o cálculo numérico. O MATLAB é capaz de compreender uma notação próxima da utilizada na literatura matemática e a linguagem M pode expressar diretamente problemas e equações, assim como traçar saídas gráficas para os *scripts* compilados pelo programa (MATHWORKS, 2019).

Para realizar os testes de SIL e a simulação de voo, utiliza-se o X-Plane 11, desenvolvido pela empresa Laminar Research. Os cenários e condições disponíveis são muito fiéis aos reais, pois são gerados a partir de dados de terrenos mapeados pela NASA em suas missões. Por sua fidelidade, a aplicação tem sido utilizada para testes de aeronaves experimentais e treino de pilotos, e recebeu a certificação da Administração Federal de Aviação dos Estados Unidos (usualmente referida como FAA) para atuar como um simulador de treino se utilizado em uma configuração de hardware adequada, que tenha suporte para processar a experiência completa (CRAIGHEAD et al., 2007). Embora possa desempenhar uma simulação de alto nível, o X-Plane também é utilizado para aplicações mais simples, principalmente para testes de novas tecnologias. Sua interface permite a customização quase completa de uma aeronave, assim como a criação de novos modelos e cenários, trazendo uma gama de possibilidades de simulação (LAMINAR RESEARCH, 2016).

Os controladores necessitam dos valores dos ângulos da aeronave para calcular a atuação necessária do veículo durante o voo, para que este seja estável. Portanto, é realizada a leitura dos ângulos e suas respectivas velocidades para utilizá-los como valores de entrada no PID. Os valores de saída são atribuídos às variáveis de controle dirigidas pelo X-Plane. Para isso, é fundamental a comunicação bidirecional e eficaz entre o agente simulador e os códigos de controle, para que ambos os programas possam transmitir e receber dados em tempo de execução. Todas as tarefas realizadas para se atingir o objetivo principal do trabalho são descritas de forma detalhada nesse capítulo.

3.1 Sistema de comunicação: MATLAB e X-Plane

Em versões anteriores do simulador de voo, a comunicação com o MATLAB era estabelecida pelo *plugin* X-Plane Connect (XPC). Em sua versão atual, o X-Plane 11 possibilita o uso do protocolo UDP de maneira nativa, viabilizando a comunicação direta com programas externos. Para tal, dispõe-se de estruturas específicas para o envio e recebimento de mensagens. Por parte do MATLAB, existe uma interface UDP onde pode ser criado um objeto do tipo *socket*, que permite a comunicação pelo protocolo UDP e suporta o envio e recebimento de

pacotes por meio de mensagens, podendo essas serem em ASCII ou binárias.

Para a criação do objeto UDP, utiliza-se o Algoritmo 3.1, onde o comando *udp* recebe como parâmetro o endereço IP e a porta de destino, sendo o MATLAB o endereço de origem.

Algoritmo 3.1 – Estrutura de criação do objeto socket.

```
1 socket = udp('127.0.0.1', 49000);
2 fopen(socket);
```

Neste caso a comunicação é feita entre dois pontos situados no mesmo ambiente, então o IP refere-se ao endereço interno do computador e a porta 49000 ao endereço padrão de recebimento do X-Plane, pois ambos os programas são executados simultaneamente em apenas uma máquina. A ação *fopen* é responsável por abrir o *socket* de comunicação entre os programas.

O X-Plane recebe e envia mensagens no formato de vetor de bytes, tendo cada requisição ou comando a sua particularidade. De maneira geral, para enviar um pacote ao destino no formato desejado, utiliza-se o comando demonstrado no Algoritmo 3.2:

Algoritmo 3.2 – Envio de uma mensagem pelo objeto socket.

```
1 fwrite(socket, message);
```

Os parâmetros do comando *fwrite* referem-se ao objeto UDP a qual será enviado o pacote e a mensagem em si. Para a leitura do pacote recebido é utilizado o comando *fread* (Algoritmo 3.3), que armazena na variável *data* os dados coletados da origem:

Algoritmo 3.3 – Recebimento de um pacote por meio do objeto de comunicação socket.

```
1 data = fread(socket);
```

Para encerrar o canal de comunicação, o *socket* é fechado, deletado e a variável é limpa, para que não ocorra indisponibilidade no sistema caso existam outras requisições para o endereço de destino, atribuído anteriormente ao objeto. Esse processo é feito por três comandos, como mostrado no Algoritmo 3.4.

Algoritmo 3.4 – Encerramento do objeto socket.

```
1 fclose(socket);
2 delete(socket);
3 clear(socket);
```

3.1.1 Requisitando dados do X-Plane

Tendo como base o objeto UDP criado no MATLAB, basta configurar a mensagem a ser enviada ao X-Plane. Os dados no simulador são mapeados em *Data Reference* (DataRef), uma estrutura identificada por uma *string* que dá acesso aos valores das variáveis da aeronave

durante o voo. Seguindo a documentação fornecida pela Laminar Research, para que o simulador retorne o valor de um determinado DataRef a mensagem deve ter o cabeçalho RREF e a estrutura representada no Algoritmo 3.5.

Algoritmo 3.5 – Estrutura da mensagem de requisição RREF.

```

1  struct dref_struct_in
2  {
3      xint dref_freq;
4      xint dref_en;
5      xchr dref_string [400];
6  };

```

O cabeçalho é um vetor de *char* de 5 bytes, terminado em caractere nulo (no caso, representado pelo número 0). A variável *dref_freq* refere-se a frequência (por segundo) do retorno da informação pedida, a *dref_en* é um código inteiro a ser enviado, que será retornado junto ao valor do DataRef para que este possa ser identificado no caso de múltiplas requisições. Por fim, o vetor de *char* *dref_string* é o nome do DataRef em si com 400 bytes. Cada caractere do vetor representa um byte, portanto, caso a *string* não preencha todas as posições, a variável deve ter as posições faltantes preenchidas com o caractere nulo. Os inteiros devem ter 4 bytes cada e a mensagem completa precisa ter 413 bytes distribuídos dessa forma para que o simulador a interprete.

Para que a mensagem seja compreendida corretamente pelo X-Plane, é necessário formatá-la antes de enviar. Para isso, é utilizado o *typecast* e os tipos nativos da linguagem M. Para que o DataRef seja preenchido com zeros até atingir as 400 posições do vetor, é formatado e armazenado na variável *dataRef_in*, onde o vetor é copiado e acrescido de zeros pela quantidade de posições faltantes.

Tendo a entrada com o tamanho e terminação corretos, a mensagem RREF foi transformada em bytes no MATLAB. As conversões do tipo *uint8* transformam cada caractere em um byte. Os inteiros passam por duas conversões, a do tipo *int32* mostra que é um único número, mas representado em 4 bytes separados. Cada linha da variável *rref* representa uma conversão dos itens da estrutura. Todo o processo descrito trata-se da variável *rref*, que traz a mensagem já formatada, como demonstrado no Algoritmo 3.6.

Algoritmo 3.6 – Formatação da mensagem no MATLAB para ser enviada ao X-Plane 11.

```

1  rref = [
2      uint8('RREF') uint8(0)
3      typecast(int32(dref_freq), 'uint8')
4      typecast(int32(dref_en), 'uint8')
5      uint8(dataRef_in)
6  ];

```

A mensagem então é enviada ao X-Plane, que retorna em apenas um pacote UDP os valores de todas as requisições, na estrutura descrita no Algoritmo 3.7.

Algoritmo 3.7 – Estrutura da mensagem de resposta RREF.

```

1  struct dref_struct_out
2  {
3      xint dref_en;
4      xflt dref_flt;
5  };

```

O pacote recebido também tem o formato de vetor de bytes, sendo os 5 primeiros caracteres o cabeçalho 'RREF0' e os 8 bytes seguintes referem-se ao valor enviado na variável *dref_en* (4 bytes) para reconhecimento e, os outros quatro, ao valor recebido em si. No caso de múltiplas requisições, a estrutura repete-se. Assim, são recebidos os valores dos ângulos de atitude da aeronave (*roll*, *pitch*), o ângulo de *sideslip* e suas velocidades angulares referentes (expressas por *p*, *q* e *r*, respectivamente), a posição da aeronave em coordenadas inercial e a intensidade da velocidade.

3.1.2 Atribuindo um valor a uma variável do X-Plane

De forma análoga à requisição de valores, para enviar valores ao X-Plane tem-se uma estrutura de mensagem a ser seguida.

No Algoritmo 3.8, *dref_path* é a *string* referente ao DataRef e a variável *var* expressa o valor que será atribuído. Seguindo o formato anterior, a mensagem deve ser composta por 509 bytes: o cabeçalho 'DREF' com terminação nula, o valor *float* em 4 bytes e o vetor de *char* com 500 posições, mas preenchido com espaços após o caractere nulo que define o fim do DataRef. Sendo assim, a formatação da mensagem no MATLAB é descrita pela variável *dref*, no Algoritmo 3.9.

Algoritmo 3.8 – Estrutura da mensagem de atribuição de valores DREF.

```

1  struct dref_struct
2  {
3      xflt var;
4      xchr dref_path [500];
5  };

```

Algoritmo 3.9 – Formatação da estrutura DREF.

```

1  dref = [
2      uint8('DREF')
3      uint8(0)
4      typecast(single(value), 'uint8')

```

```
5     uint8(dref_path)
6     uint8(0)
7     uint8(padding)
8 ];
```

A variável *padding* é um vetor de espaços, para preencher a mensagem até que ela obtenha o tamanho de 509 bytes. A mensagem é enviada pelo *socket* e o efeito do valor passado pode ser visto imediatamente no simulador, no comportamento da aeronave durante o voo.

3.2 Seguimento de caminho

Com a finalidade de que a simulação apresente maior semelhança ao uso real de um VANT, foi desenvolvido um algoritmo de seguimento de caminho para os experimentos e comparação das estratégias. O caminho implementado para a comparação das estruturas de controle usando testes SIL tem o formato de um quadrado e a função de seguimento trabalha com a orientação da aeronave por meio de quatro *waypoints*. Os *waypoints* são pontos fixos definidos pelo sistema de coordenadas local inercial NEU (*North-East-Up*) cuja ligação por retas determina o caminho a ser seguido. Logo, esses pontos são um local de checagem, pois quando um ponto é atingido pela aeronave, é realizada uma manobra de curva para que a mesma siga em linha reta, em direção a outro ponto. Sendo assim, a aeronave segue em um voo reto até encontrar o ponto que a fará mudar de posição, pois o caminho designado é uma linha reta que interliga esses quatro *waypoints*.

Para iniciar a função de seguimento de caminho foi necessário definir a posição inicial da aeronave, visto que sem essa configuração o simulador a inicia em locais aleatórios. Então, foram escritos os DataRefs *local_x*, *local_y* e *local_z*, para que o veículo fosse carregado sempre na mesma coordenada e o caminho pudesse ser replicado.

Com a comunicação entre o X-Plane e o MATLAB estabelecida e a posição inicial definida, durante a simulação são lidas as coordenadas do GPS. Para pegar a posição da aeronave é preciso transformar essas coordenadas para a unidade do sistema de coordenadas local inercial que lê como norte o eixo x, leste o eixo y e o eixo z para cima. Essa conversão é feita por meio da função *geodetic2enu* do MATLAB, que recebe os valores da posição atual medidos de latitude, longitude e altitude e os valores de um ponto de referência de latitude, longitude e altitude, além de um sistema que determina o cálculo dos valores georreferenciados, que foi determinado como o elipsóide de referência WGS84 ([NATIONAL IMAGERY AND MAPPING AGENCY, 2000](#)).

Essa posição transcrita é passada para a função *path_following*, que executa de fato o seguimento de caminho e define os valores desejados para os ângulos de atitude, usando o algoritmo Lookahead ([PELIZER; SILVA; BRANCO, 2017](#)) e manobras de curvas coordenadas. Esses valores dos ângulos *roll* e *yaw* são passados como referência para os controladores PID

que compõem as duas estratégias usadas nesse trabalho. O ângulo *roll*, especificamente, passa por uma mudança brusca durante a simulação e isso é refletido em seus valores de referência. Isso acontece porque a aeronave precisa mudar completamente a sua direção ao atingir um *waypoint* e, para fazer essa manobra corretamente, é utilizado o atuador *aileron* (que comanda o ângulo *roll*) para forçar a aeronave a mudar de posição e realizar a curva, enquanto o ângulo *pitch* varia, basicamente, com o erro da altitude alcançada durante o voo.

3.3 Testes de *Software In the Loop*

Estabelecida a troca de informações entre o X-Plane e o MATLAB e com as estratégias de controle implementadas, é possível, então, controlar o VANT. Os dados provenientes do simulador são utilizados para o cálculo dos valores de entrada para os PID utilizados. Os valores de saída, que são os resultados dos controladores, são enviados para a aeronave e definem o seu comportamento.

3.3.1 Experimentos de simulação

Os *scripts* de simulação foram feitos através de laços (*loops*) de iteração, onde os controladores são executados em sequência para que o controle da aeronave seja estabelecido. Para criar essas funções iterativas, primeiro estabeleceu-se a conexão via UDP e foi realizada a leitura dos valores dos ângulos de atitude vindos do simulador, por meio do processo explicado na Subseção 3.1.

Em ambas as estratégias de controle implementadas, o *script* de simulação é iniciado da mesma forma, para que estejam em condições iniciais idênticas em ambos os experimentos. Isso é feito por meio de um arquivo chamado de *Situation*, disponibilizado pelo X-Plane. Este arquivo permite salvar, literalmente, situações de voo como o cenário, condições temporais, posição da aeronave e demais informações atreladas a um voo. No caso, utilizou-se o arquivo "Chicago Landmarks", que é dado pelo simulador e inicia o veículo sempre no mesmo cenário, já em etapa de voo em cruzeiro. O voo em cruzeiro, por sua vez, compreende a situação de voo em que a aeronave encontra-se estável, em voo horizontal, mantendo sua velocidade e altitude constantes. Trata-se de um movimento em equilíbrio, onde é possível visualizar melhor o desempenho dos controladores pois representa uma condição de voo estável. Além de definir uma condição inicial de voo, a posição geográfica e física do corpo da aeronave também são definidas, pois notou-se que mesmo com o voo em cruzeiro aplicado, muitas vezes o veículo iniciava a simulação com empinamento ou algum desajuste nos ângulos de atitude. Assim, foi criada a função *init_xplane_position*, onde são escritos os DataRefs para que a aeronave comece cada *loop* exatamente com os mesmos ângulos em todas as replicações do experimento, garantindo que estes sejam exatamente iguais para cada estratégia, para uma comparação justa.

Para a estratégia de PID para posição, o *script* carrega o arquivo de *Situation* e, com

a aeronave devidamente iniciada em voo cruzeiro, são lidos os valores iniciais dos ângulos *roll*, *pitch*, *sideslip* e de velocidade da aeronave Cessna Skyhawk. Esses valores são passados em forma de vetor para a função *set_controllers_horizontal*, que é responsável por calcular os erros que são a entrada do controlador, e tem em seu corpo as variáveis referentes ao ganho de cada parte do PID. É uma função de definição dos controladores.

Sendo assim, essa função têm como parâmetro a variável *delta* de tempo (o valor é expresso em segundos e descreve o período da execução do controlador), os valores dos ângulos e velocidade iniciais, um vetor que define os valores máximos de deflexão para cada superfície de controle e para a atuação do motor (para que a saída do controlador possa ser passada diretamente aos atuadores) e os ângulos iniciais desejados. Para este último parâmetro, foi passado como referência o valor 2° para *roll* e *pitch* e 0° para o *sideslip*. Todos os ângulos são dados em graus, mas convertidos para radiano, que é a unidade de medida utilizada pelo controlador. Essa conversão é feita pela função nativa *deg2rad* da linguagem M.

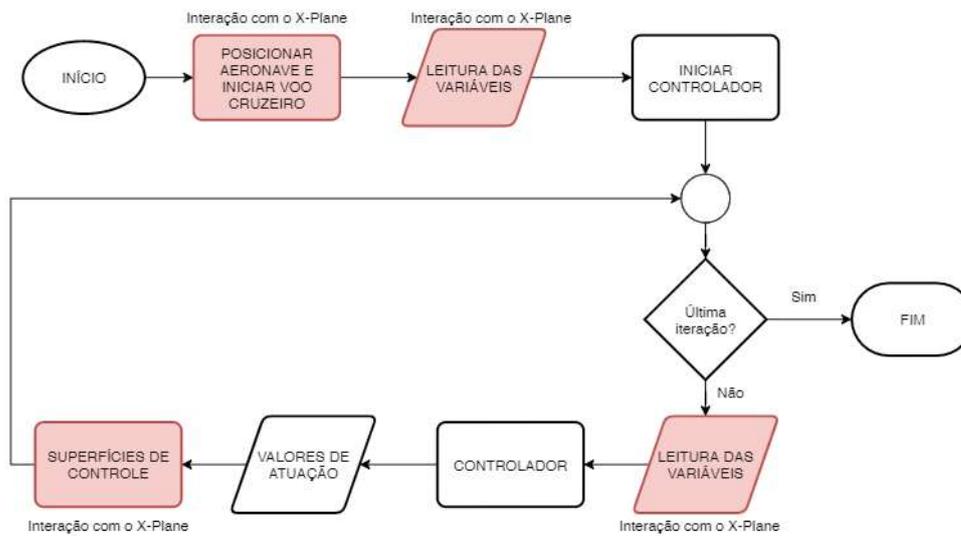
Com essas definições e valores, é iniciado o *loop* de simulação de fato, em uma estrutura de repetição. Para cada iteração do laço, é realizada a leitura das variáveis e feita uma validação, para certificar que não sejam valores vazios. Esses valores recebidos são passados para a função de controle que recebe como parâmetro os valores desejados para os ângulos de atitude, os ângulos lidos no início do laço, a velocidade da aeronave e a variável *Controllers*, que é a estrutura usada para guardar as informações dos erros dos ângulos e outro parâmetros que definem o funcionamento do controlador PID discretizado, como foi mostrado nas Figuras 5 e 6.

Dessa maneira, o controlador PID calcula (como descrito na Seção 2.2) o esforço necessário dos atuadores da aeronave para que os ângulos de atitude e a velocidade sejam estáveis. Os valores de saída são dados em intensidade de deflexão para a superfície de controle e são passados diretamente para o X-Plane por meio da escrita dos DataRefs correspondentes ao *rudder*, *throttle*, *aileron* e *elevator* que são responsáveis, respectivamente, pelo controle de *sideslip*, velocidade, *roll* e *pitch*.

Essa estrutura se repete até que o laço de repetição acabe, e todo esse processo é realizado para cada uma das iterações. A função de simulação foi desenhada para ser modular, de forma que sua implementação seja a mesma em todas as utilizações, para garantir a fidelidade dos testes. A Figura 8 representa o funcionamento do *script* de simulação.

3.3.2 Otimização com o Algoritmo Genético para o ajuste de ganhos

O Algoritmo Genético (AG) é um algoritmo evolutivo de otimização que faz o uso de técnicas da ciência biológica para encontrar os melhores valores dado um problema. O algoritmo começa criando uma população inicial, que pode ser aleatória ou criada a partir de pontos iniciais, quando o problema já possui soluções conhecidas e é desejado que a otimização tenha início a partir desses pontos. Então define-se o número de populações desejado e o AG cria uma sequência de novas populações. A cada etapa são utilizados os indivíduos da geração

Figura 8 – Fluxograma do *script* de simulação utilizado nos experimentos.

Fonte: Elaborada pela autora.

atual para criar a próxima, onde os melhores indivíduos dessa geração são considerados para a subsequente, para que sejam obtidos melhores indivíduos do que a população anterior, criando uma sequência evolutiva (MITCHELL, 1998; MATHWORKS, 2020).

Neste trabalho, busca-se os valores ideais para os ganhos Proporcional, Integral e Derivativo (K_P , K_I e K_D) dos controladores PID utilizados para cada uma das estratégias abordadas. Para isso, o Algoritmo Genético é utilizado para otimizar os ganhos, baseado em duas situações: voo em cruzeiro livre, onde a aeronave apenas segue um trajeto reto, e o voo em cruzeiro com seguimento de caminho, como descrito na Seção 3.2. O objetivo é comparar qual conjunto de ganhos possui melhor resultado para cada estratégia, para que estas sejam comparadas dentro dos melhores resultados obtidos para cada uma. Portanto, foram realizadas quatro otimizações, utilizado o PID para posição com e sem trajetória e o PID para posição e velocidade nas mesmas situações.

Os *scripts* de simulação do PID para posição com e sem seguimento de caminho e o do PID para posição e velocidade, também com e sem seguimento de caminho, foram adaptados para funções que recebem como parâmetro a variável x , um vetor que expressa os ganhos de cada controlador PID. Assim, a cada geração o valor é alterado e o desempenho é mensurado e levado em consideração na nova rodada. Dessa forma, o Algoritmo Genético pode encontrar os indivíduos mais aptos (que convergem para um resultado mais satisfatório).

Para que a otimização considere os erros obtidos a cada geração do algoritmo, a função de cada estratégia calcula a diferença entre os ângulos e velocidade obtidos na simulação e seus valores desejados. Esses erros passam por uma normalização, a partir do valor obtido pela norma de cada um deles, visto que os erros dispõem um formato de vetor durante a aplicação. A norma 2 do erro faz a média do quadrado de cada um dos valores obtidos no *loop* e, a partir da norma, os erros são normalizados. A partir da normalização, é calculada a função custo:

uma função que traz a soma desses valores e a devolve para o AG.

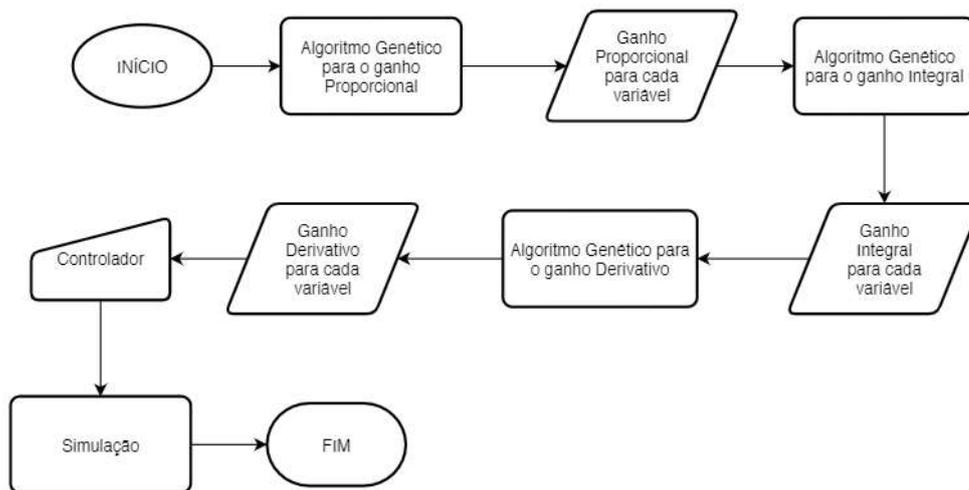
Durante os primeiros testes na implementação da otimização com Algoritmo Genético, notou-se que muitas vezes os valores obtidos para os ganhos não eram suficientes para o PID funcionar efetivamente e manter a estabilidade do voo na simulação. Quando isso ocorre, a aeronave perde o controle e cai. O ato de cair faz com que o tempo da otimização aumente, porém não gera indícios ao algoritmo de que não está obtendo um resultado satisfatório.

Sendo assim, foi realizada também a leitura do DataRef que indica se houve uma queda e, ao ser identificado que a aeronave caiu durante a execução da otimização, é adicionada uma penalidade de 100 pontos. Essa penalidade é multiplicada pela soma das normalizações dos erros, elevando o custo final da função caso tenha ocorrido algum desgaste. Caso contrário, a saída da função mantém-se apenas como a soma dos erros normalizados, expressando o custo real da aplicação.

O procedimento foi realizado separadamente para cada ganho do PID para posição, primeiro para o *script* sem seguimento de caminho e, depois, para o que possui essa função. Da mesma forma, foi feito para o PID de posição e velocidade, com e sem seguimento de caminho. Primeiro foi realizada a obtenção dos conjuntos de ganhos com a otimização, para depois realizar a comparação das estratégias de controle, onde cada uma utilizou o conjunto que apresentou melhor resultado em seu uso. A otimização consiste em utilizar o AG para o ganho K_P primeiro, enquanto os outros ganhos são nulos, exceto para o K_I do controlador do *pitch* que deve ser igual a um para compensar um erro permanente observado nas simulações. Ao finalizar a otimização para o ganho Proporcional, o algoritmo retorna o melhor valor encontrado para cada controlador, então esses valores são passados ao controlador e utilizados para a otimização do ganho Integral. Portanto, o ganho K_I é obtido já utilizando os valores obtidos para K_P .

De maneira análoga, o algoritmo seleciona os melhores valores encontrados para o ganho Integral e estes são utilizados para a otimização do ganho Derivativo, junto aos valores K_P obtidos na primeira parte do processo. Em resumo, inicia-se com a otimização do ganho Proporcional, os valores obtidos são utilizados na otimização do ganho Integral. Então, o valor obtido para K_I a partir de K_P , é utilizado, em conjunto com este para obter o ganho K_D , como exemplificado na Figura 9.

Figura 9 – Processo de obtenção dos ganhos utilizando o Algoritmo Genético.



Fonte: Elaborada pela autora.

4 ANÁLISE DOS RESULTADOS

Neste capítulo encontram-se os resultados obtidos durante os experimentos descritos no Capítulo 3. Para todas as comparações aqui descritas, para ambas estratégias de controle, foi utilizada a simulação com seguimento de caminho, por representar um cenário mais real e mais completo, com maior complexidade.

4.1 Ganhos obtidos para a estratégia de controle de posição

Como apresentado no Capítulo 3, cada estratégia obteve dois conjuntos de ganhos (Proporcional, Integral e Derivativo) por meio da otimização com o Algoritmo Genético. Para isso, foram definidos os parâmetros necessários para o funcionamento do algoritmo, sendo eles o número da população para cada rodada e o número de gerações. Foi estabelecido o valor de 10 indivíduos para a população e de 10 gerações para cada execução do algoritmo, ao considerar que um maior número de população ou geração resultaria em uma otimização extremamente longa. Com esses valores utilizados, o tempo médio para cada execução do AG foi de 4 horas. A grande quantidade de tempo para realizar a otimização se dá ao fato de que é necessário executar uma parte do voo da aeronave para obter os valores de atuação. Assim, para cada rodada do Algoritmo Genético é realizada uma simulação de voo, ocasionando em um alto tempo de execução.

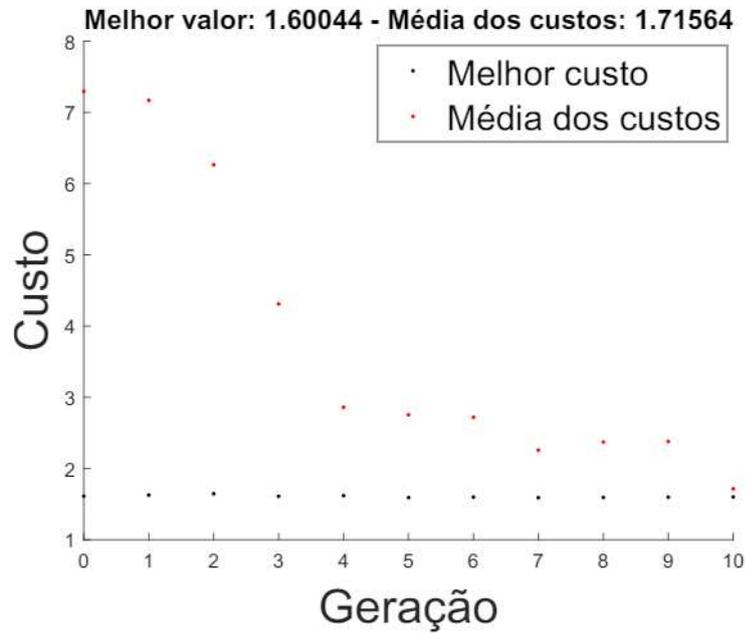
O primeiro conjunto foi obtido a partir da simulação de voo da aeronave com o PID de posição sem a função de seguimento de caminho. Nessa otimização, é possível avaliar como os controladores minimizam o erro das variáveis de controle sem considerar a variação das entradas, o que ocorre realmente em um voo com caminho definido. As funções de custo de cada geração do AG para os ganhos K_P , K_I e K_D podem ser vistas nas Figuras 10, 11 e 12.

A otimização dos ganhos também foi feita para o controle de posição com seguimento de caminho, onde foram utilizados valores de entrada para a população inicial do algoritmo, visto que o esforço necessário para a atuação da aeronave no seguimento de caminho é muito alto. Com esses valores preliminares, o resultado da otimização pode ser mais satisfatório. Esses valores de entrada foram os ganhos obtidos de forma manual, ou seja, encontrados de modo empírico – por tentativa e erro –, onde foram testados valores até que o controlador pudesse atuar de forma adequada.

Os valores de custo do algoritmo para cada geração da otimização em busca dos ganhos para os controladores PID de posição utilizando a simulação com seguimento de caminho estão ilustrados nas Figuras 13, 14 e 15.

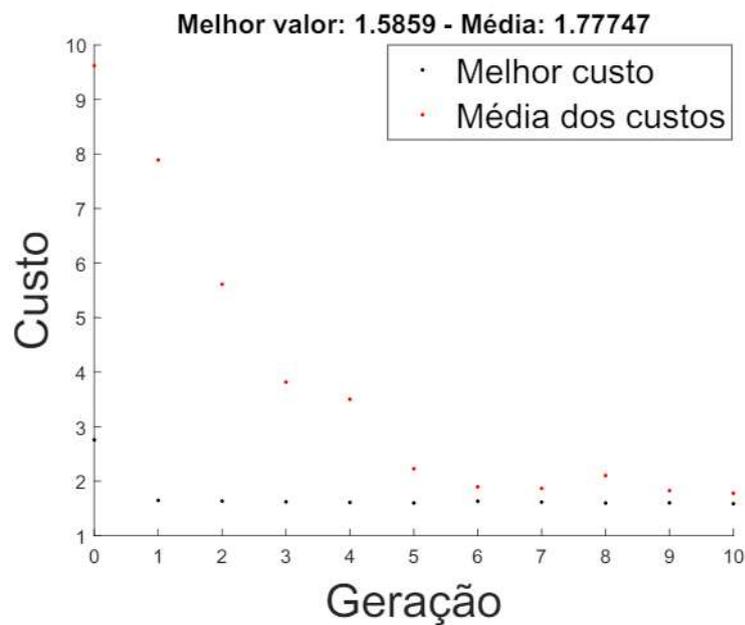
Para eleger qual conjunto de ganhos possui o melhor desempenho para a estratégia abordada foi utilizada a análise estatística. Para gerar as métricas, o experimento de simulação com trajeto foi repetido 20 vezes, a fim de que os valores gerados fossem suficientes para a

Figura 10 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição com Algoritmo Genético, sem seguimento de caminho.



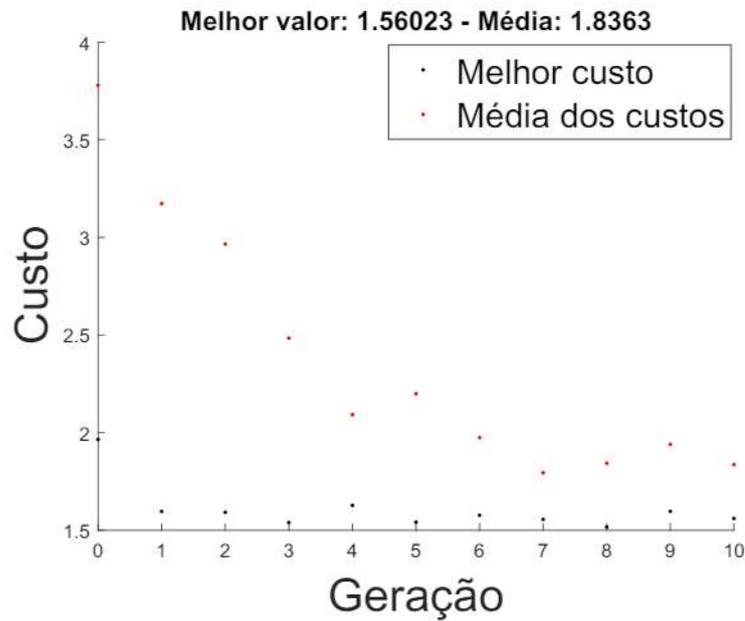
Fonte: Elaborada pela autora.

Figura 11 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição com Algoritmo Genético, sem seguimento de caminho.



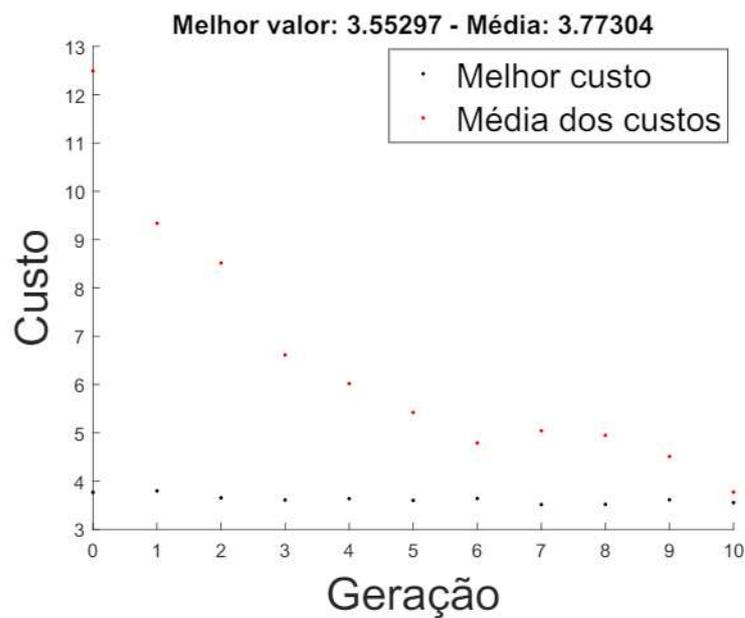
Fonte: Elaborada pela autora.

Figura 12 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição com Algoritmo Genético, sem seguimento de caminho.



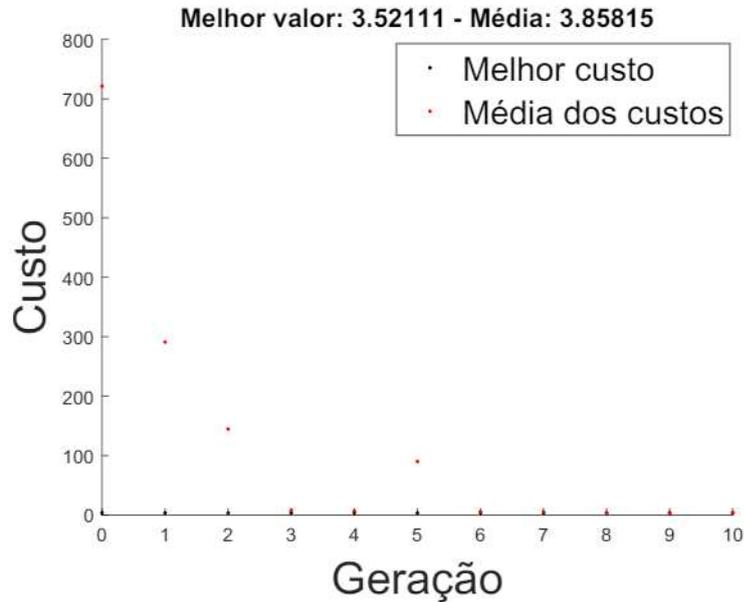
Fonte: Elaborada pela autora.

Figura 13 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição com Algoritmo Genético, com seguimento de caminho.



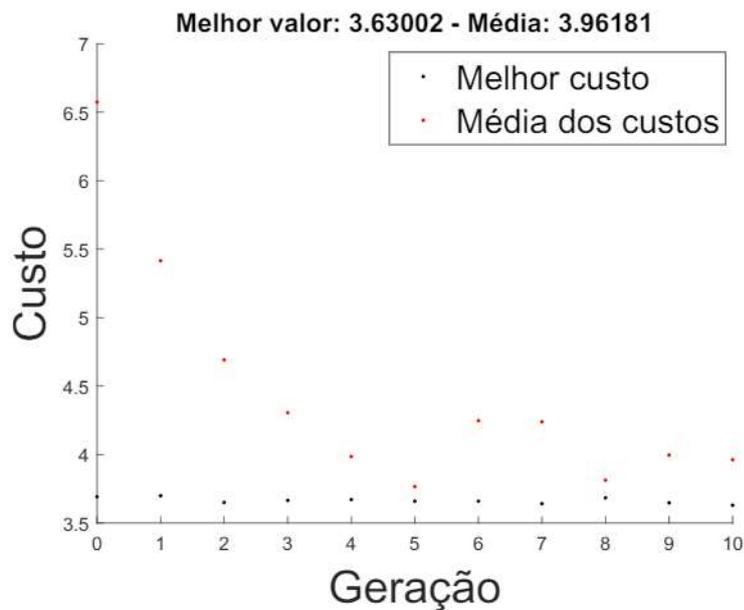
Fonte: Elaborada pela autora.

Figura 14 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição com Algoritmo Genético, com seguimento de caminho.



Fonte: Elaborada pela autora.

Figura 15 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição com Algoritmo Genético, com seguimento de caminho.



Fonte: Elaborada pela autora.

análise. Em cada replicação, foi extraída a norma do erro de cada variável de controle, sendo esses valores essenciais para este método. A partir dos valores encontrados, foi calculada a média para cada variável e também o desvio padrão para o intervalo. Todos os cálculos foram realizados no Microsoft Excel e suas fórmulas próprias. Os dados são provenientes de um *script* de comparação formulado em linguagem M, que gerou um arquivo CSV exportado diretamente do MATLAB.

Com o desvio padrão definido, foi calculado o intervalo de confiança para cada um. Esse intervalo é calculado com base no número de experimentos, desvio padrão desses experimentos e um nível de confiança de 95%. Isso demonstra que a probabilidade de uma amostra de experimento estar dentro do intervalo de confiança é compatível a esse valor, assumindo que há uma distribuição normal. Portanto, o experimento é replicado para que as medidas sejam validadas, visto que um único experimento pode não ser representativo pois as medidas podem sofrer distúrbios (JAIN, 1990).

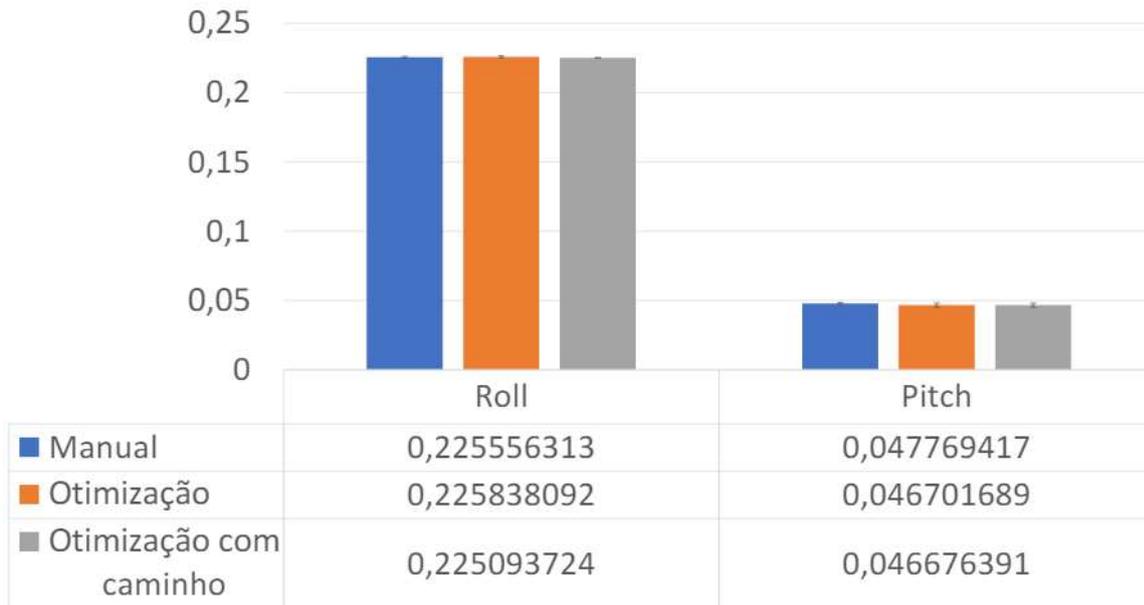
A análise em si é realizada por meio de gráficos de barra, onde são desenhadas as médias de cada variável e os intervalos de confiança como barra de erro para cada uma delas. Assim, é estimado que se os intervalos de confiança não se sobrepõem, as médias são consideradas estatisticamente diferentes e são, portanto, uma maior do que a outra. Se os intervalos de confiança sobrepõem-se significativamente e as médias encontram-se dentro desses intervalos, elas são estatisticamente iguais para o nível de confiança definido. Caso os intervalos se sobreponham, mas a média não se encontra dentro do intervalo de confiança do outro, não há conclusões. Nesse caso, é necessário aumentar o número de repetições para obter medidas mais significativas e palpáveis (JAIN, 1990).

As médias dos ângulos *roll* e *pitch* apresentaram valores semelhantes para todos os casos analisados, onde os intervalos de confiança se sobrepõem, como é observado na Figura 16. Isso indica que a diferença entre os valores de referência e os valores medidos durante os experimentos foi baixa e todos são considerados estatisticamente iguais, mostrando que o controlador conseguiu atingir um resultado satisfatório com os três conjuntos de ganhos.

Para a média das normas dos erros do ângulo de *sideslip*, representada na Figura 17, é notado que os dois primeiros conjuntos de ganhos possuem grande diferença em relação ao terceiro. Este comportamento observado está relacionado à tentativa do controlador em melhorar o valor para o ângulo de *roll* cuja normalização apresenta o maior valor. Sendo assim, houve o aumento no erro do ângulo de *sideslip* como forma de compensação para as demais variáveis, gerando grande instabilidade nos valores apresentados. Os ganhos manuais e da otimização são equivalentes e estão em intervalos de confiança que se abrangem, mas os ganhos da otimização com seguimento de caminho são elevados. Isso implica que a média dos dois primeiros são estatisticamente iguais e, a do último conjunto de ganhos, é diferente e maior. Com pouca diferença, o conjunto que se destaca pelo desempenho é o obtido pelo Algoritmo Genético sem trajeto.

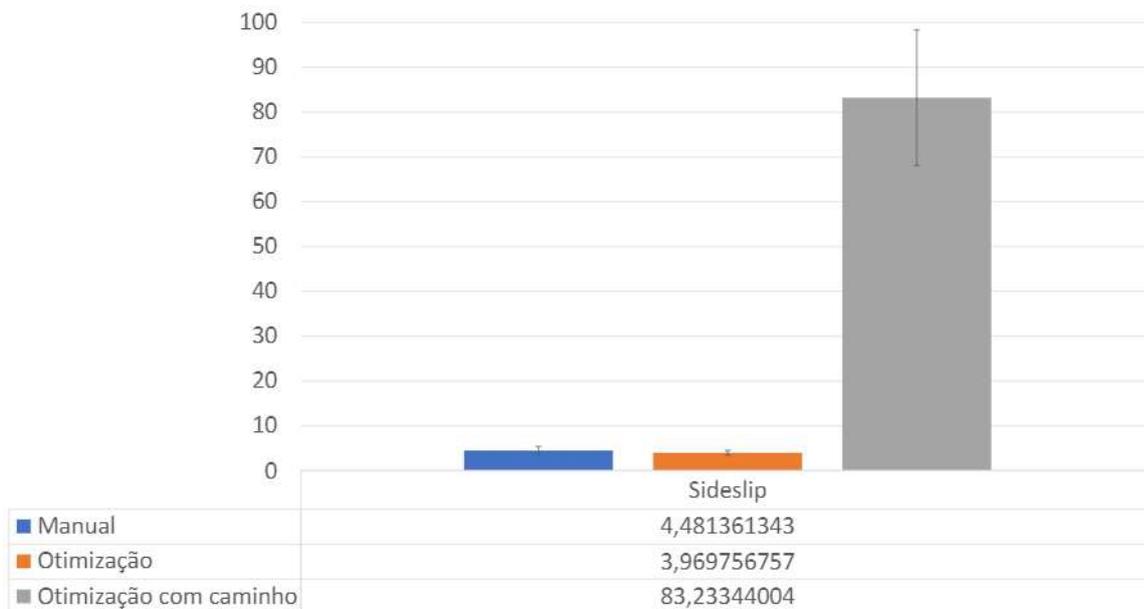
Tratando da velocidade, cujo gráfico está representado na Figura 18, observa-se

Figura 16 – Gráfico de comparação entre os conjuntos de ganhos para as variáveis *roll* e *pitch* do controlador PID para posição.



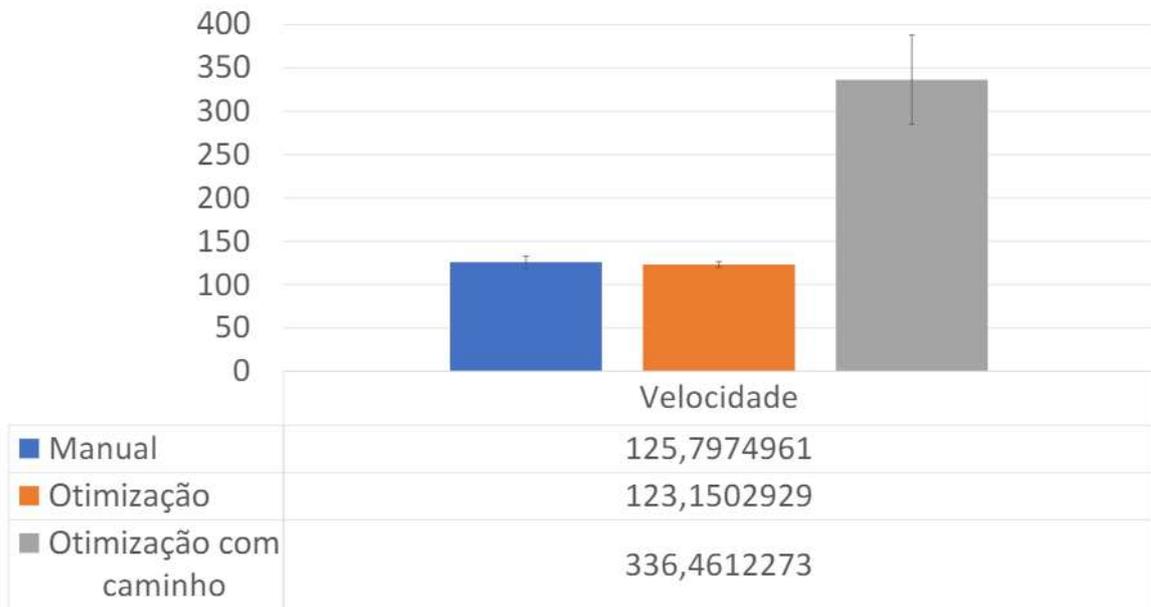
Fonte: Elaborada pela autora.

Figura 17 – Gráfico de comparação entre os conjuntos de ganhos para o ângulo de *sideslip* do controlador PID para posição.



Fonte: Elaborada pela autora.

Figura 18 – Gráfico de comparação entre os conjuntos de ganhos para a variável de velocidade do controlador PID para posição



Fonte: Elaborada pela autora.

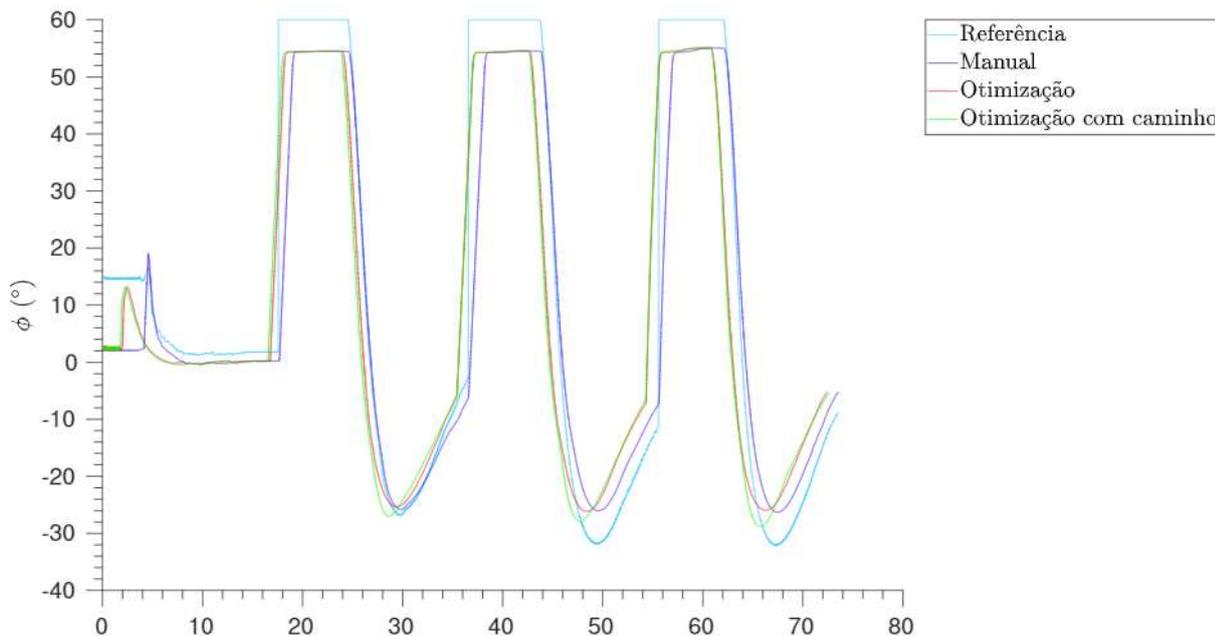
o mesmo comportamento descrito para o *sideslip*. Os dois primeiros conjuntos apresentam resultados semelhantes e são equivalentes, enquanto o último apresenta os piores resultados. Ressalta-se que quanto menor a média, menor é a diferença entre os valores desejados para a variável e os valores obtidos pela simulação, o que é desejado para um bom desempenho do controlador. Sendo assim, novamente os ganhos conseguidos pela otimização sem seguimento de caminho proporcionaram a menor média, portanto, o melhor desempenho do controlador.

A fim de visualizar mais precisamente o comportamento da aeronave para cada um desses conjuntos, foram desenhados gráficos da trajetória e do comportamento das variáveis de estado em relação aos valores desejados. Para a comparação entre os conjuntos de ganhos foi excluído o ganho Derivativo. Durante os experimentos, notou-se que a presença de K_D aumenta o efeito de *chattering* nas variáveis controladas em todos os casos. Esse efeito ocasiona em oscilações de alta frequência na saída do controlador, causando maior instabilidade no valor das variáveis e no resultado desejado (KNOSPE, 2006). A exclusão desses ganhos pôde maximizar a atuação da aeronave, visto que diminui a interferência e oscilação nas saídas do controlador para as superfícies de controle do veículo.

Para a estratégia de controle de posição, como já visto anteriormente por meio da análise estatística, é possível notar que os ganhos manuais e os ganhos da otimização simples possuem valores semelhantes e, portanto, comportamentos semelhantes. As Figuras 19, 20 e 21 mostram os valores dos ângulos *roll*, *pitch* e *sideslip*, respectivamente. Os valores medidos durante o experimento para os dois primeiros conjuntos acompanham os valores de referência,

embora não se igualem fielmente a este em nenhum dos casos. O terceiro conjunto, os ganhos obtidos pela otimização com seguimento de caminho, apresentam maior divergência entre o valor medido e o valor de referência, além da presença do efeito de *chattering* (oscilação no sinal).

Figura 19 – Comportamento de cada conjunto de ganho para o ângulo de *roll* em relação aos valores de referência na estratégia de controle para posição.



Fonte: Elaborada pela autora.

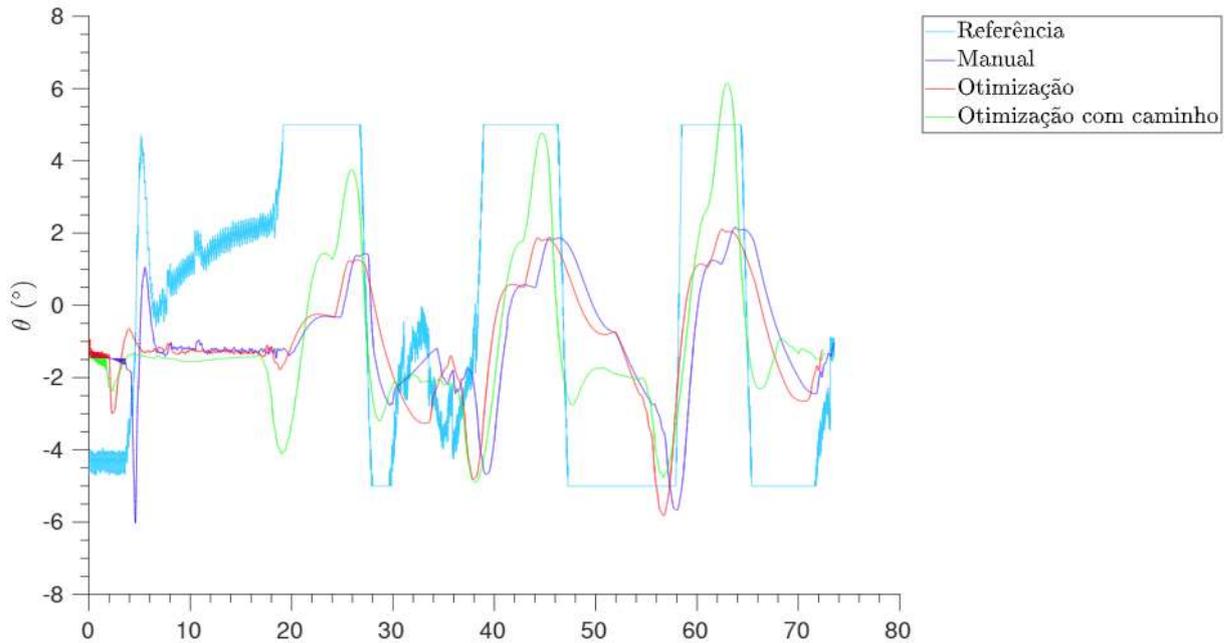
A Figura 22 traz o comportamento de cada grupo em relação ao valor desejado para a velocidade, definido como 70 m/s. Todos os conjuntos demonstram dificuldade para controlar e estabilizar a variável, embora com uma margem de erro baixa. Isso indica que o esforço do motor tem foco em manter a altitude da aeronave, ocasionando em oscilações na velocidade.

Por fim, tem-se a comparação de cada grupo durante o seguimento de caminho. A Figura 23 traz a visualização do trajeto desejado em 3D e o comportamento físico da aeronave durante a simulação para cada tipo de ganho. Para essa estratégia, todos os conjuntos de ganhos puderam finalizar o trajeto designado, mesmo com pequenas oscilações. Isso mostra que todos esses valores proporcionam uma atuação adequada ao controlador, com pouca diferença entre si.

4.2 Ganhos obtidos para a estratégia de controle de posição e velocidade

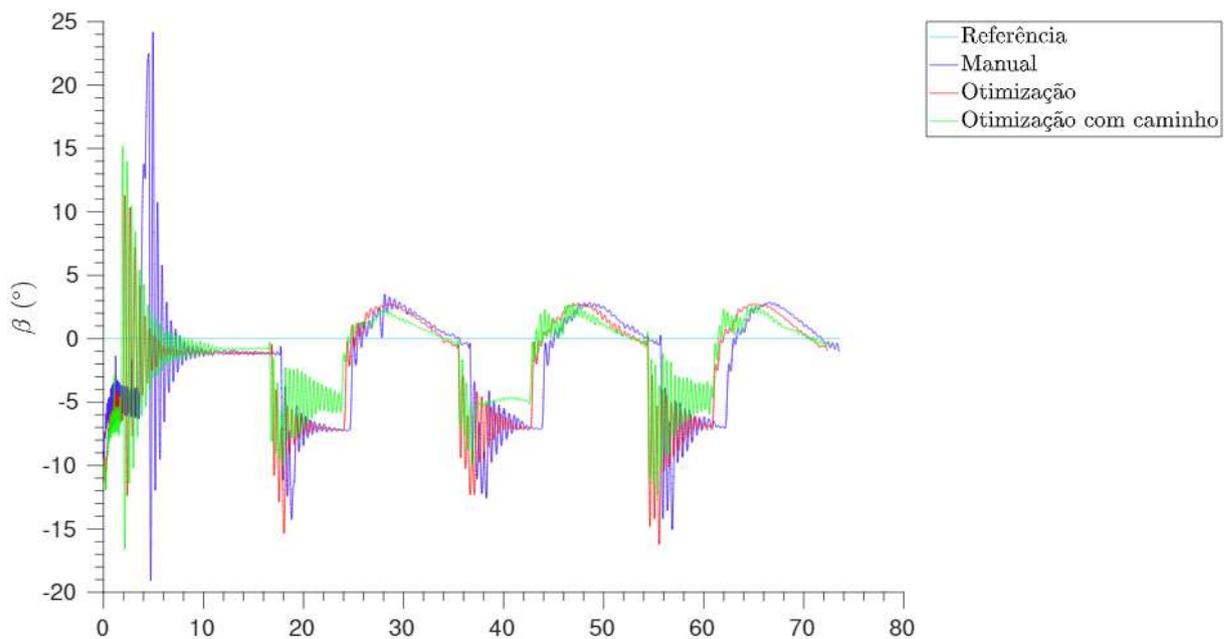
Para conseguir os conjuntos de ganhos com e sem seguimento de caminho para a estratégia de posição e velocidade, o procedimento foi exatamente igual ao realizado para o PID para posição e velocidade, descrito na seção anterior (4.1). Os valores das funções de custo para o AG sem seguimento de caminho encontram-se nas Figuras 24, 25 e 26.

Figura 20 – Comportamento de cada conjunto de ganhos para o ângulo de *pitch* em relação aos valores de referência na estratégia de controle para posição.



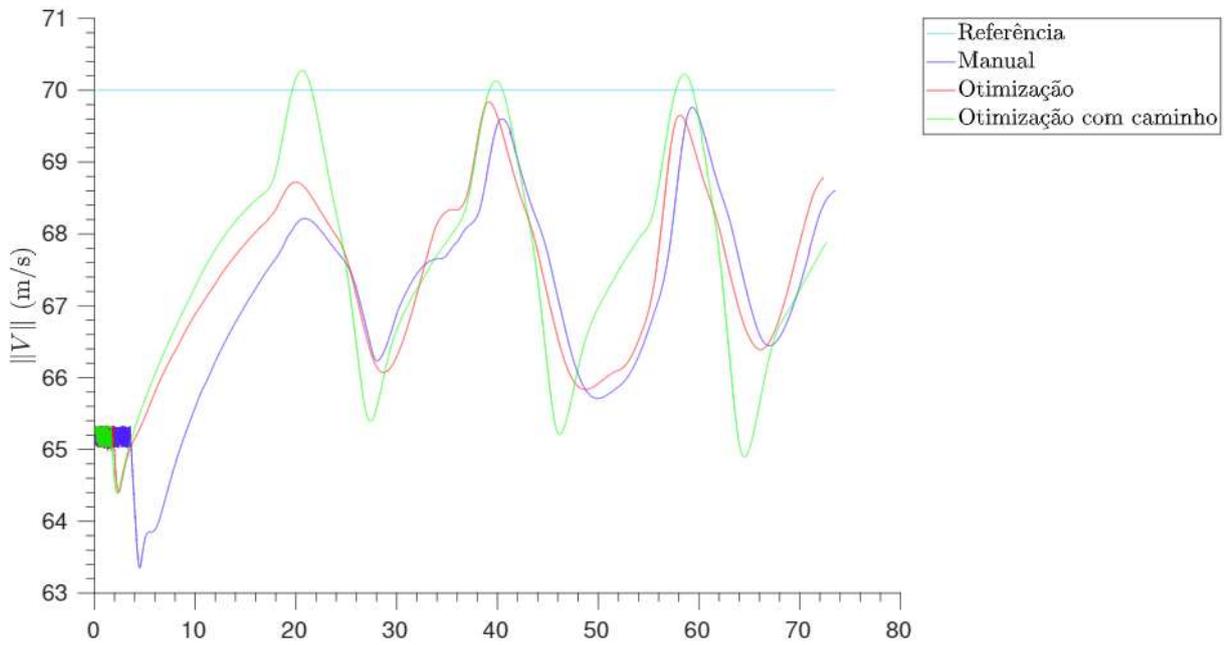
Fonte: Elaborada pela autora.

Figura 21 – Comportamento de cada conjunto de ganhos para o ângulo de *sideslip* em relação aos valores de referência na estratégia de controle para posição.



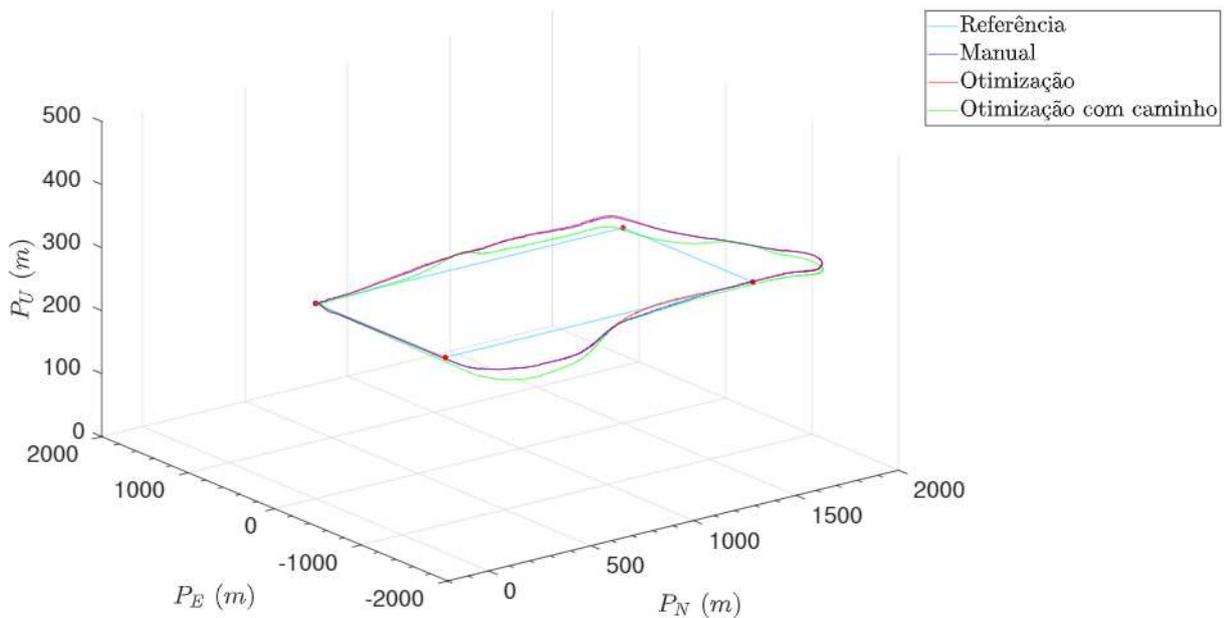
Fonte: Elaborada pela autora.

Figura 22 – Comportamento de cada conjunto de ganhos para velocidade em relação ao valor de referência na estratégia de controle para posição.



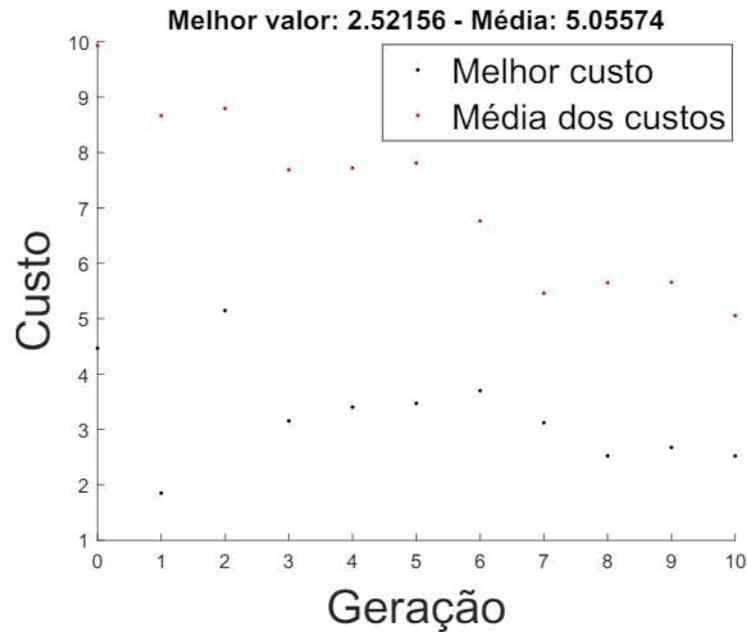
Fonte: Elaborada pela autora.

Figura 23 – Comportamento de cada conjunto de ganhos durante o seguimento de caminho para a estratégia de controle para posição.



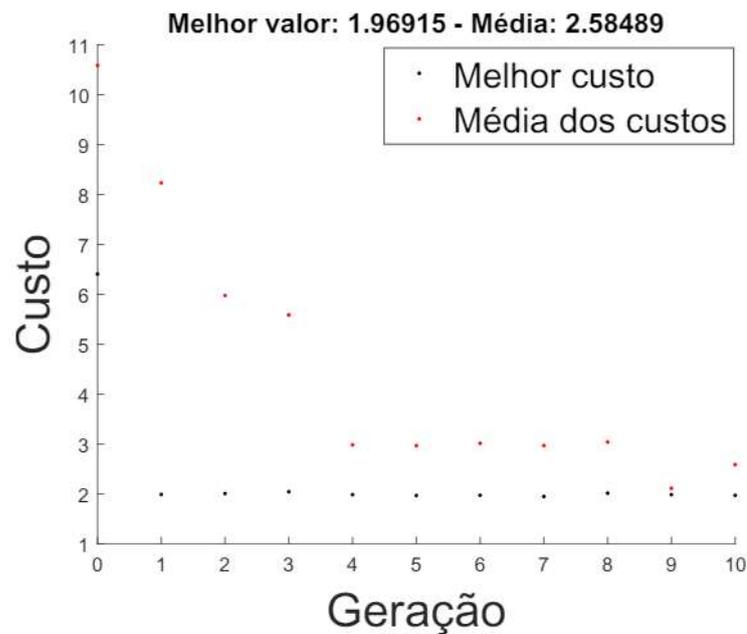
Fonte: Elaborada pela autora.

Figura 24 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição e velocidade com Algoritmo Genético, sem seguimento de caminho.



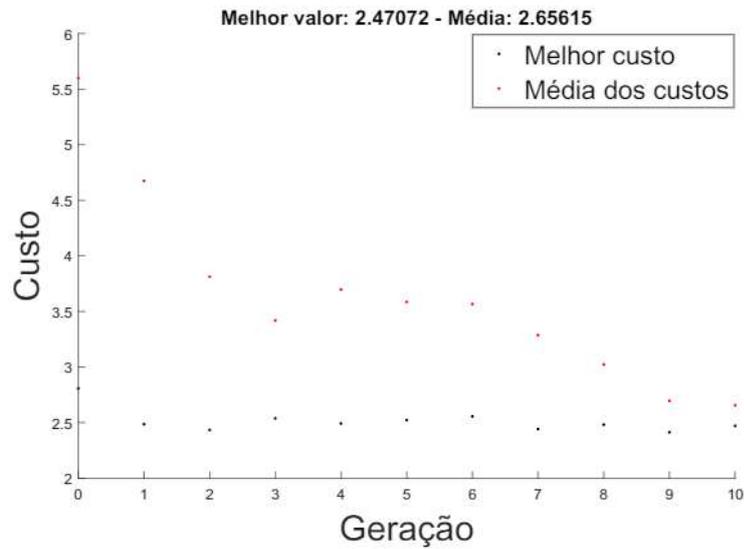
Fonte: Elaborada pela autora.

Figura 25 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição e velocidade com Algoritmo Genético, sem seguimento de caminho.



Fonte: Elaborada pela autora.

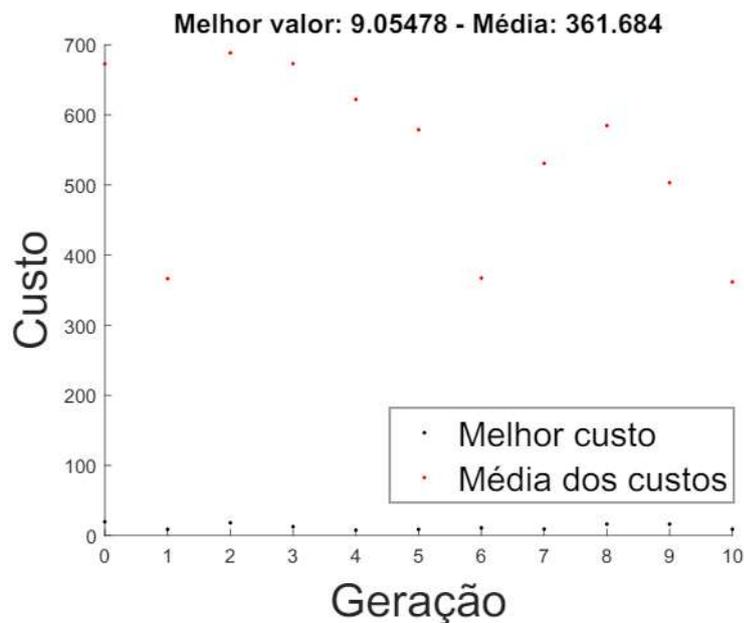
Figura 26 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição e velocidade com Algoritmo Genético, sem seguimento de caminho.



Fonte: Elaborada pela autora.

As funções de custo para o algoritmo com o seguimento de caminho, utilizando também como valores de entrada os ganhos obtidos manualmente, estão expressas nas Figuras 27, 28 e 29.

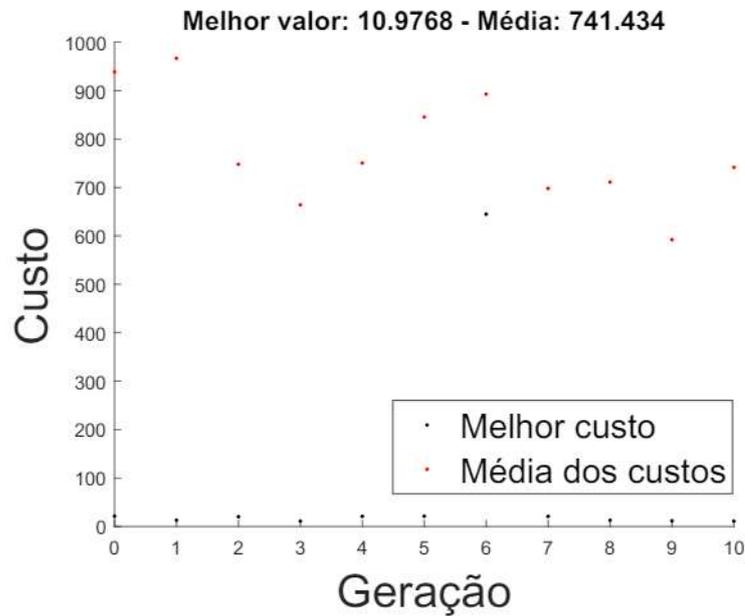
Figura 27 – Funções de custo para o ajuste de ganhos proporcionais da estratégia de controle de posição e velocidade com Algoritmo Genético, com seguimento de caminho.



Fonte: Elaborada pela autora.

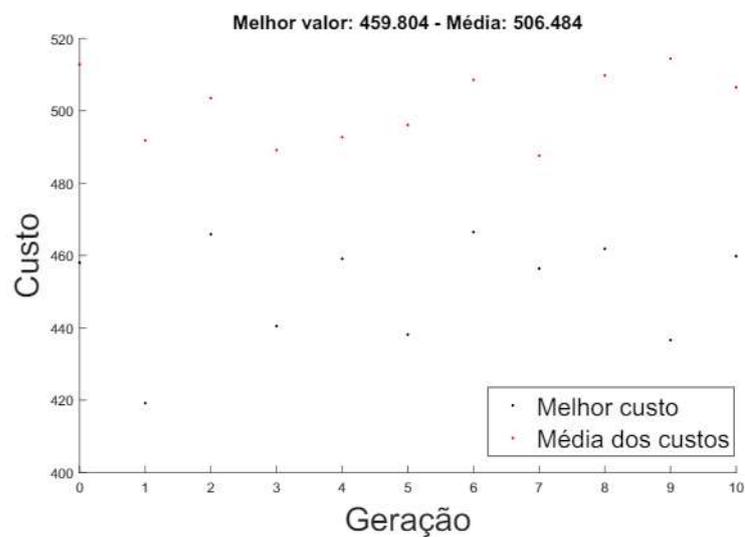
Na execução do Algoritmo Genético, para todos os casos e estratégias, notou-se um comportamento anormal, contrário à literatura. Por definição, o AG utiliza os melhores

Figura 28 – Funções de custo para o ajuste de ganhos integrais da estratégia de controle de posição e velocidade com Algoritmo Genético, com seguimento de caminho.



Fonte: Elaborada pela autora.

Figura 29 – Funções de custo para o ajuste de ganhos derivativos da estratégia de controle de posição e velocidade com Algoritmo Genético, com seguimento de caminho.



Fonte: Elaborada pela autora.

indivíduos de uma geração (chamados de elite) como indivíduos da população subsequente (MATHWORKS, 2020). Como visto nas Figuras 12, 13, 15, 24, 25, 26, 27, 28 e 29, onde os pontos pretos representam a melhor função custo de cada geração, em alguns casos as gerações subsequentes apresentam pontos mais altos, quando a função de custo da geração seguinte deveria ser menor, nunca maior do que a resultante da geração anterior (MITCHELL, 1998).

Ao investigar a ocorrência, foram descartadas hipóteses sobre inconsistências nas simulações, visto que esse tipo de risco foi mitigado com a função *init_xplane_position* e a modularização das funções de simulação, que garantem que todas as replicações ocorram em situações iniciais idênticas. Foram listados, então, cenários que não podem ser controlados e que podem causar esse comportamento no Algoritmo Genético.

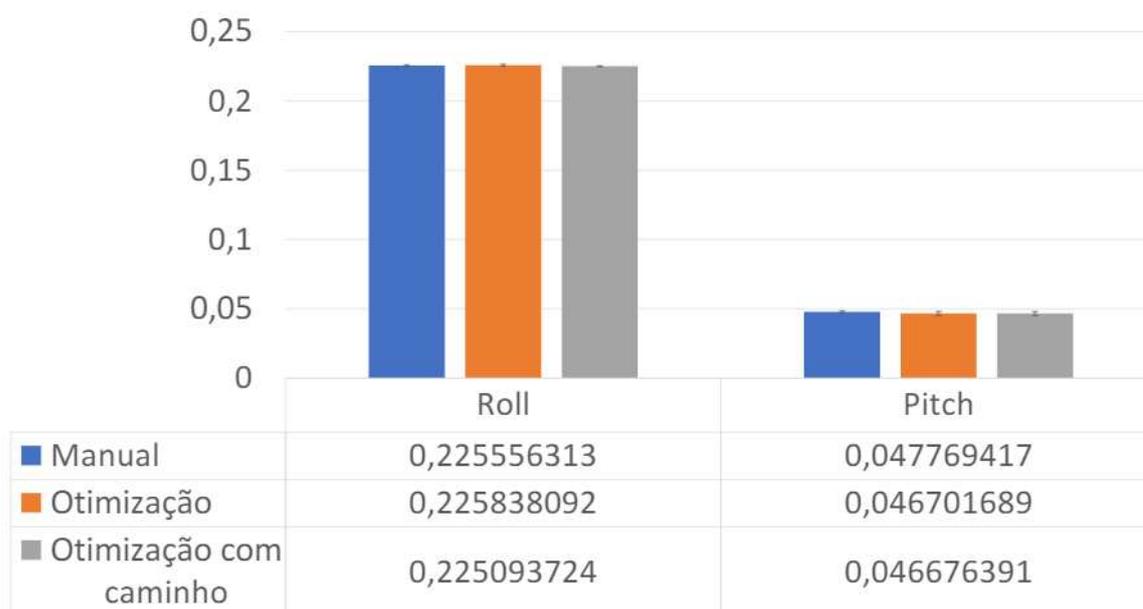
Uma das primeiras observações trata das instabilidades do simulador de voo, pois as ações dos controladores podem ocorrer em tempos diferentes, visto que são realizados em tempo de simulação e não há como garantir que este seja o mesmo para todos os casos. O tempo de simulação não se passa em tempo real e está diretamente ligado ao desempenho do computador que está sendo utilizado e, também, distúrbios do próprio experimento que podem refletir em mudanças nas forças aerodinâmicas da aeronave, justificando uma função de custo mais alta pelo esforço maior.

O segundo cenário é relacionado à instabilidade da comunicação com o UDP, pois algumas mensagens podem ser perdidas durante a transmissão. Essa é uma característica conhecida do protocolo, pois o UDP é considerado, pela literatura, como não confiável por não ser orientado à conexão, então não existe uma garantia de que o pacote de dados seja entregue corretamente ao destino (TANENBAUM, 2003). Podem ocorrer entregas fora de ordem, então todos os *scripts* possuem uma checagem das informações pelo número do DataRef e checagem por soma, que garante uma quantidade mínima de informações para o funcionamento dos controladores e atribuição correta de valores. Ainda assim, não há como atestar que todas as informações sejam recebidas e que estejam completas, o que pode ocasionar em custos diferentes para cada geração do algoritmo, pois não há garantia de que as informações recebidas para cada geração sejam exatamente as mesmas. A questão de processamento e desempenho do computador durante a simulação também pode ter influência neste caso, pois podem ocorrer atrasos nas entregas e falta de sincronia entre os programas utilizados, devido ao esforço do processador central para manter ambos funcionando simultaneamente.

Ainda assim, mesmo que o comportamento das funções de custo não tenha sido o esperado, os resultados obtidos pelo AG mostraram-se, em sua maioria, satisfatórios, como pode ser observado nos gráficos de comparação entre os conjuntos de ganhos e também pelo melhor valor da função de custo demonstrada nos gráficos da otimização.

Assim como para a primeira estratégia, o melhor conjunto de ganhos para o controlador de posição e velocidade foi definido por meio da análise estatística, com as médias das normas dos erros e um intervalo de confiança de 95% de uma amostra de 20 replicações. Para essa estratégia as médias das normas dos erros *roll* e *pitch* apresentaram, novamente, valores baixos

Figura 30 – Gráfico de comparação entre os conjuntos de ganhos para as variáveis *roll*, *pitch* do controlador PID para posição e velocidade.



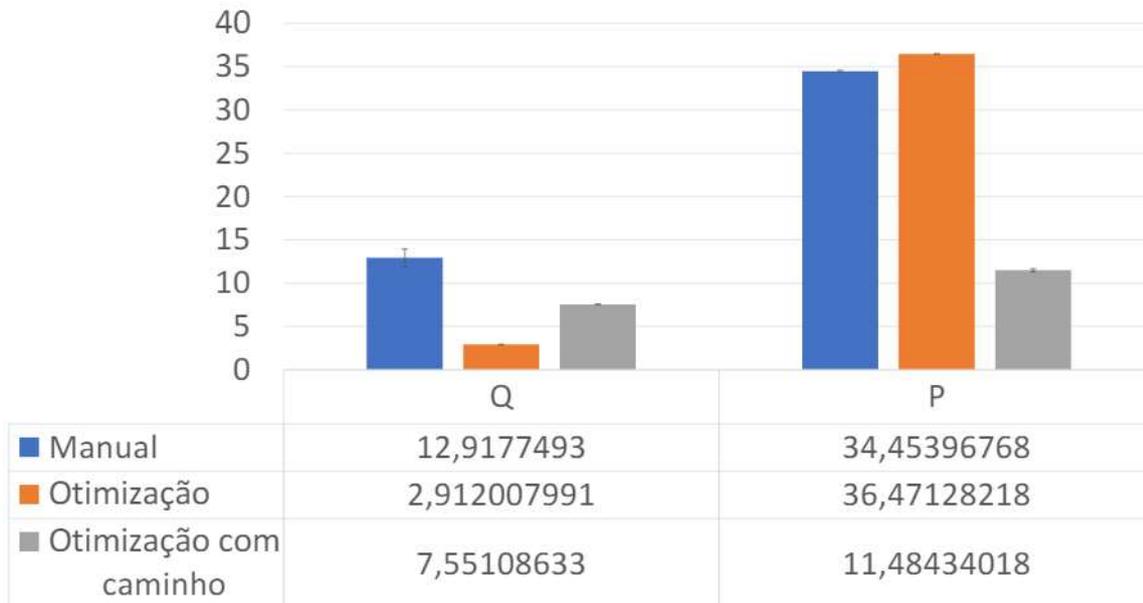
Fonte: Elaborada pela autora.

e intervalos de confiança sobrepostos, evidenciando que essas medidas são estatisticamente iguais para o intervalo definido, como considerado na Figura 30. Para as velocidades angulares, Q e P, tem-se valores divergentes, onde os intervalos de confiança não são sobrepostos e, portanto, as médias podem ser consideradas diferentes (vide Figura 31). Sendo assim, para a variável Q o ganho obtido pela otimização mais simples, sem seguimento de caminho, obteve a melhor média. Já para P a otimização com seguimento de caminho trouxe melhores resultados.

Para a variável de *sideslip* (Figura 32), as médias também são diferentes. Mais uma vez, os ganhos manuais não trouxeram valores satisfatórios e o conjunto do Algoritmo Genético sem seguimento de caminho tem a menor média entre os analisados. Tratando-se da velocidade, representada na Figura 33, tem-se novamente um alto valor vindo dos ganhos manuais. Para esse caso, o conjunto obtido pelo AG com seguimento de caminho representa a menor média, visto que as médias são estatisticamente diferentes.

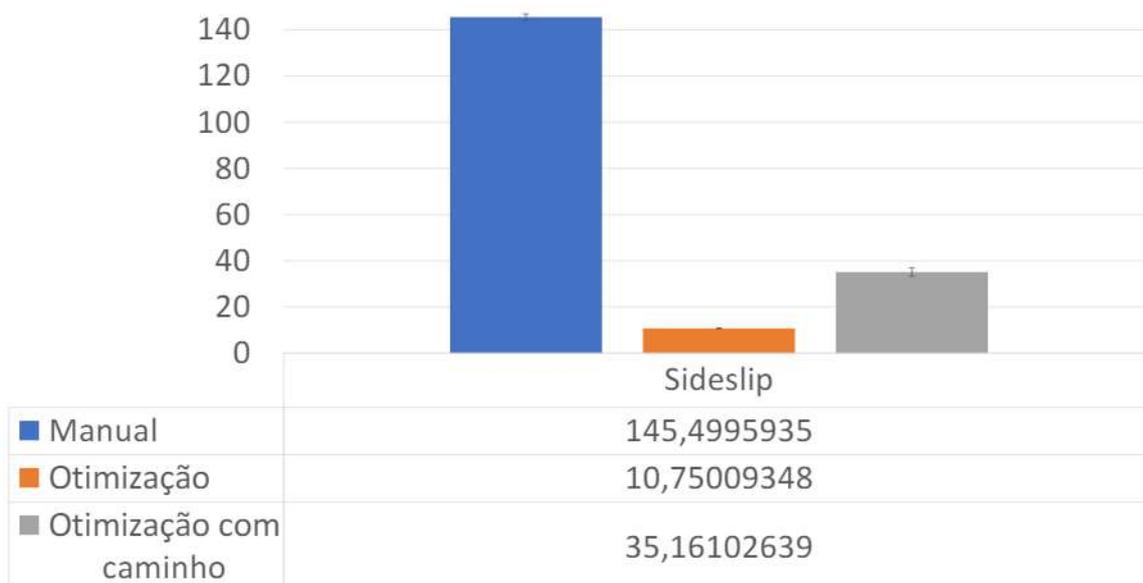
É observado que os ganhos alcançados com os dois tipos de otimização (com e sem seguimento de caminho) apresentam, de maneira geral, bons resultados. Na maioria dos casos, ambos apresentam valores semelhantes, com pouca diferença, embora para a velocidade angular estes apresentem valores bem desiguais. Para essas comparações, também foram desconsiderados os ganhos K_D para diminuir o efeito *chattering* e, por este comportamento visualizado nas velocidades angulares, vê-se que o único ganho que de fato aconteceu foi em relação ao β (*sideslip*). Como os outros parâmetros não obtiveram uma melhoria significativa nos valores, o ângulo de *sideslip* pode ter influenciado muito a função custo para compensar os

Figura 31 – Gráfico de comparação entre os conjuntos de ganhos para as velocidades angulares Q e P do controlador PID para posição e velocidade.



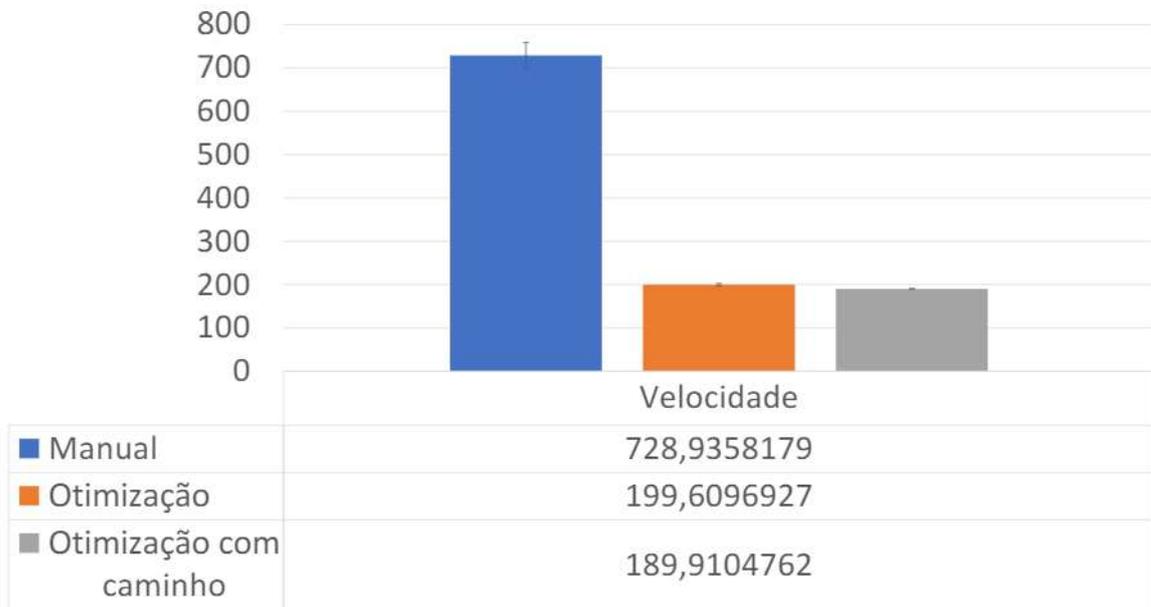
Fonte: Elaborada pela autora.

Figura 32 – Gráfico de comparação entre os conjuntos de ganhos para o ângulo de *sideslip* do controlador PID para posição e velocidade.



Fonte: Elaborada pela autora.

Figura 33 – Gráfico de comparação entre os conjuntos de ganhos para a variável de velocidade do controlador PID para posição e velocidade.



Fonte: Elaborada pela autora.

demais.

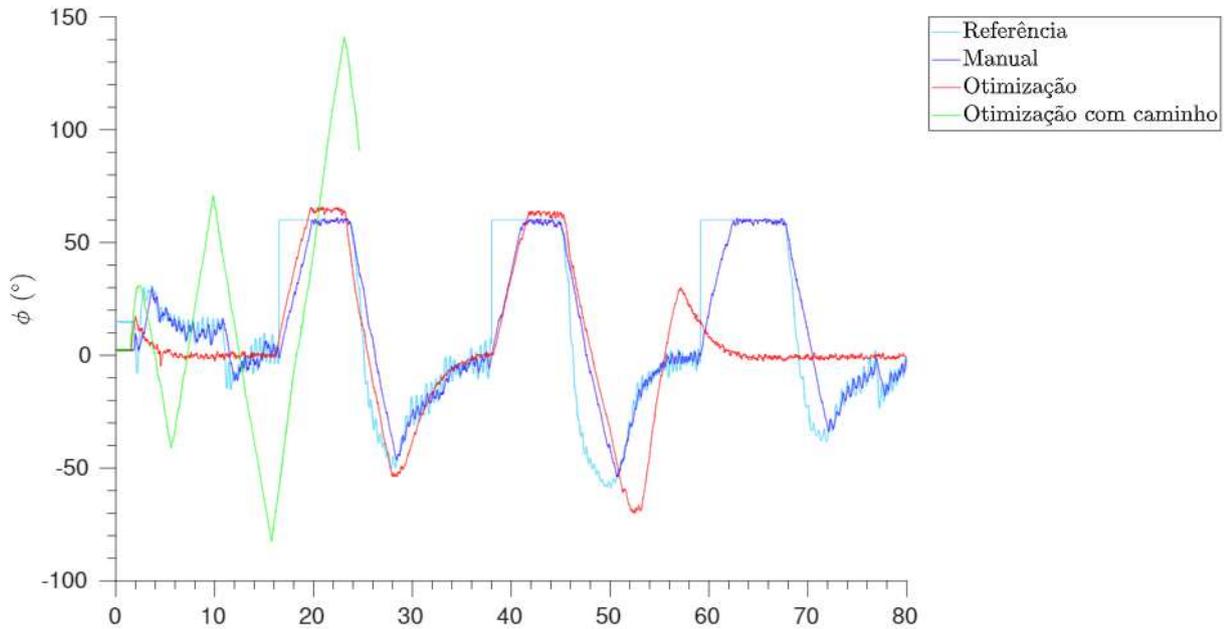
Sendo assim, o ângulo β é um fator de decisão para a escolha do melhor conjunto de ganhos. Como mostra a Figura 32, os ganhos da otimização sem seguimento de caminho apresentaram a menor média – lembrando que quanto menor a média, menor é a diferença entre o valor lido durante a simulação e o valor estimado – e, também, um bom desempenho geral. Dessa forma, para a estratégia de controle de posição e velocidade esses ganhos podem proporcionar resultados mais desejáveis.

Da mesma maneira ao que foi realizado para a primeira estratégia, tem-se também os gráficos de estado de cada ângulo, velocidade e trajeto para cada um dos conjuntos de ganhos. Nesse caso, vê-se que todos os grupos obtiveram comportamento diferente aos valores desejados. Para a otimização com caminho, a aeronave não consegue finalizar o trajeto após alcançar o primeiro *waypoint*, por isso os valores nos gráficos para esse conjunto de ganhos se encerra em um tempo anterior aos outros.

Para os ângulos de atitude *roll*, *pitch* e para o ângulo de *sideslip*, Figuras 34, 35 e 36 respectivamente, é possível notar uma alta instabilidade no comportamento dos valores ao longo do tempo de simulação. Para a velocidade, variável representada na Figura 37 há uma maior uniformidade e menor oscilação dos valores, indicando que essa estratégia possui um melhor controle da velocidade do veículo em relação ao PID apenas para posição.

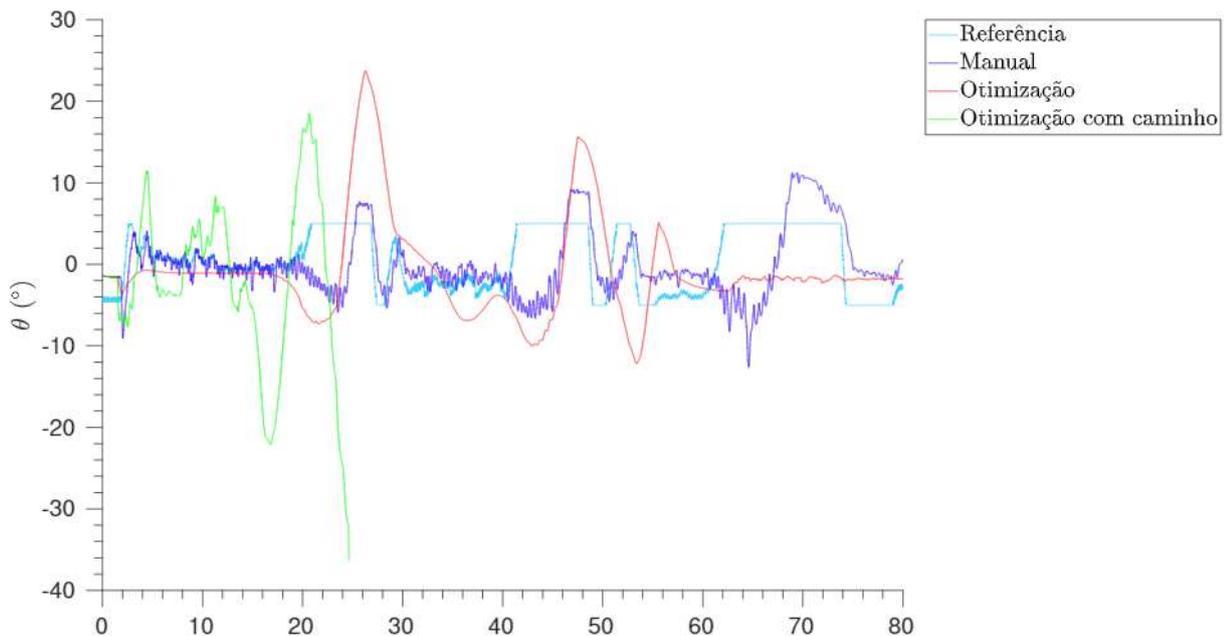
As instabilidades vistas nas variáveis de estado refletem no desempenho dos ganhos para que a aeronave consiga concluir o trajeto definido na simulação. Para esta estratégia de

Figura 34 – Comportamento de cada conjunto de ganho para o ângulo de *roll* em relação aos valores de referência na estratégia de controle para posição e velocidade.



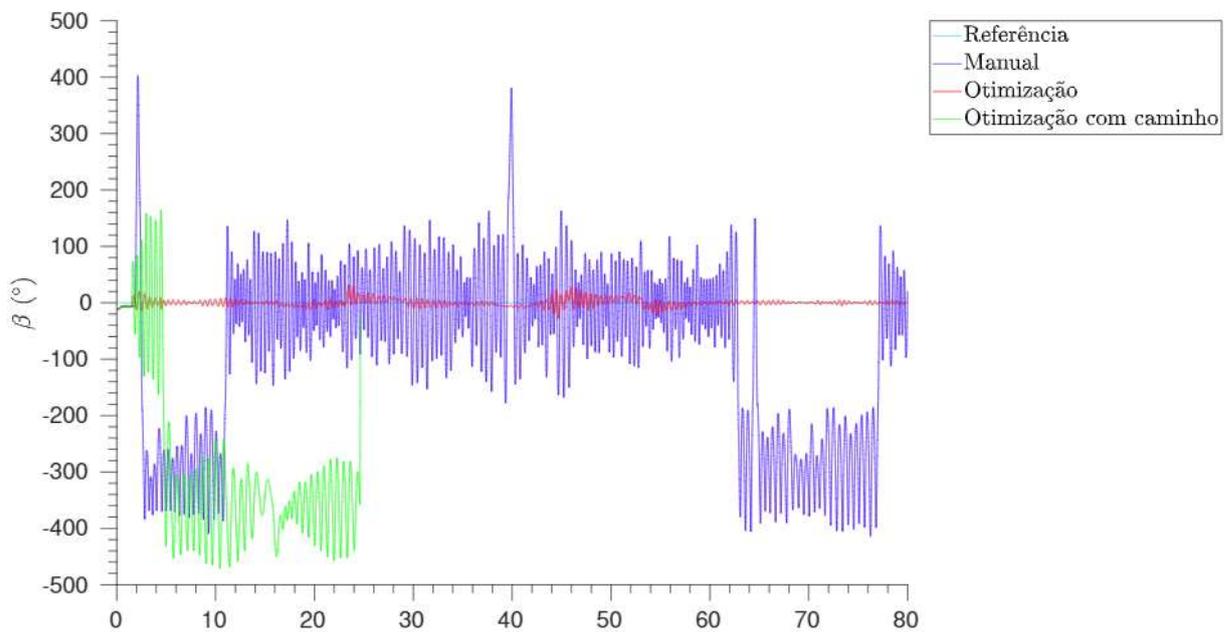
Fonte: Elaborada pela autora.

Figura 35 – Comportamento de cada conjunto de ganhos para o ângulo de *pitch* em relação aos valores de referência na estratégia de controle para posição e velocidade.



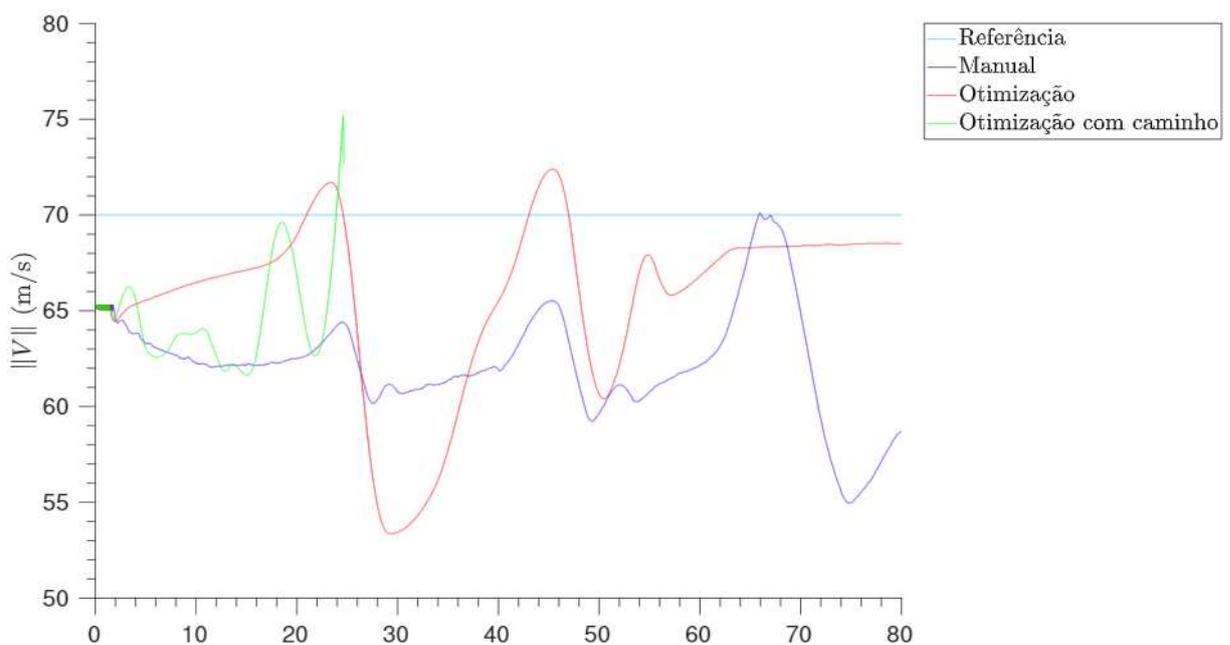
Fonte: Elaborada pela autora.

Figura 36 – Comportamento de cada conjunto de ganhos para o ângulo de *sideslip* em relação aos valores de referência na estratégia de controle para posição e velocidade.



Fonte: Elaborada pela autora.

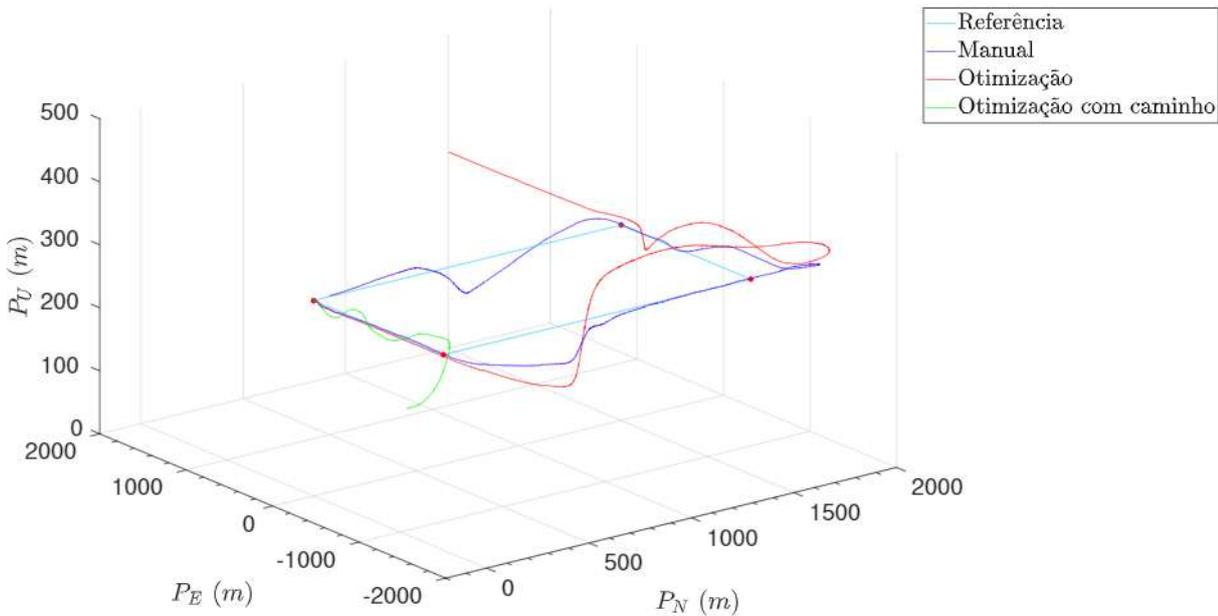
Figura 37 – Comportamento de cada conjunto de ganhos para velocidade em relação ao valor de referência na estratégia de controle para posição e velocidade.



Fonte: Elaborada pela autora.

controle, para posição e velocidade, nenhum dos conjuntos forneceu recursos suficientes para que o veículo concluísse percurso, como observa-se na Figura 38. Isso indica que nenhum dos ganhos foi suficiente para sustentar a atuação do controlador por completo, embora tenha se chegado bem próximo do objetivo.

Figura 38 – Comportamento de cada conjunto de ganhos durante o seguimento de caminho para a estratégia de controle para posição e velocidade.



Fonte: Elaborada pela autora.

Ressalta-se que todos os procedimentos realizados para a estratégia de controle de posição e de controle de posição e velocidade foram os mesmos, desde a posição inicial da aeronave na simulação, condições de voo, trajeto, até a otimização e obtenção dos ganhos. Foram fornecidas condições e recursos iguais para que a comparação pudesse ser feita de maneira justa.

4.3 Comparação entre as estratégias de controle

Após definir qual conjunto de ganhos foi o melhor para cada estratégia, realizou-se, então, a comparação entre ambas. Para as duas estratégias, o melhor conjunto de ganhos foi aquele obtido pela otimização simples. Os valores para cada variável e cada termo do controlador encontram-se na Tabela 1 e 2, representando os valores utilizados para o PID para posição e o controlador para posição e velocidade, respectivamente. Buscou-se os melhores resultados de otimização para que ambos os controladores possam ser comparados em seu melhor desempenho obtido dentro deste experimento.

A comparação foi feita para cada estratégia utilizando a simulação com seguimento de caminho, ilustrando uma situação mais realista de voo, onde o esforço da aeronave é maior e há mudanças em sua posição. Para cada um dos controladores foram gerados gráficos que indicam

Tabela 1 – Ganhos utilizados para a estratégia de controle para posição.

Variável	K_P	K_I	K_D
<i>Roll</i>	5,00	0	0
<i>Pitch</i>	3,72	1	0
<i>Sideslip</i>	0,50	0	0
Velocidade	15,10	0	0

Tabela 2 – Ganhos utilizados para a estratégia de controle para posição e velocidade.

Variável	K_P	K_I	K_D
<i>Roll</i>	18,2675	1,0000	0
<i>Pitch</i>	19,2683	3,0000	0
<i>Sideslip</i>	0,4642	0,1250	0
Velocidade	16,5000	1,1250	0
Q	0,7514	0,2500	0
P	10,0453	0	0

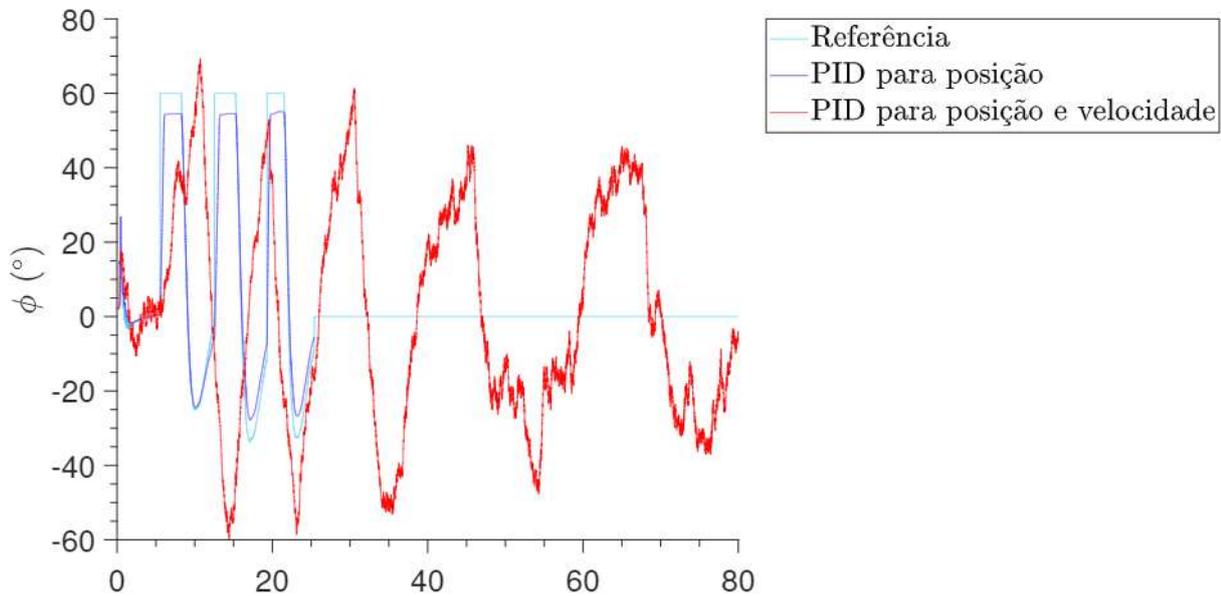
os valores desejados para os ângulos e velocidade, passados como referência aos controladores, e os valores lidos durante a simulação. Assim, é possível observar o comportamento para cada uma das variáveis de controle e visualizar se os valores são próximos. Em relação ao percurso estipulado, para melhor visualização tem-se um gráfico tridimensional que representa o comportamento da aeronave em relação ao trajeto definido. Com esse gráfico é possível observar como o corpo físico da aeronave se comportou durante o voo. Todas as imagens foram geradas no MATLAB.

Para os ângulos *roll* e *pitch* (Figuras 39 e 40, respectivamente), nota-se que o PID para posição segue próximo aos valores de referência, enquanto a outra estratégia oscila bastante durante o percurso. É visto também que a definição para a primeira estratégia acaba antes da segunda, isso ocorre porque cada simulação acontece em um tempo diferente, portanto a primeira estratégia concluiu o trajeto em um tempo de simulação menor que o PID para posição e velocidade.

Para o *sideslip* (Figura 41) ambas estratégias apresentaram trepidações nos valores. O controlador PID para posição e velocidade apresentou um comportamento muito diferente do desejado, enquanto a outra estratégia para posição apresentou valores mais coerentes e com menor erro em relação ao esperado. Para a variável de velocidade, representada na Figura 42, observa-se a que ambas as estratégias não conseguem manter a aceleração estável. Entretanto, novamente o PID para posição apresenta a atuação mais próxima ao estipulado, com momentos onde consegue até mesmo alcançar o valor de referência, mas não consegue mantê-lo.

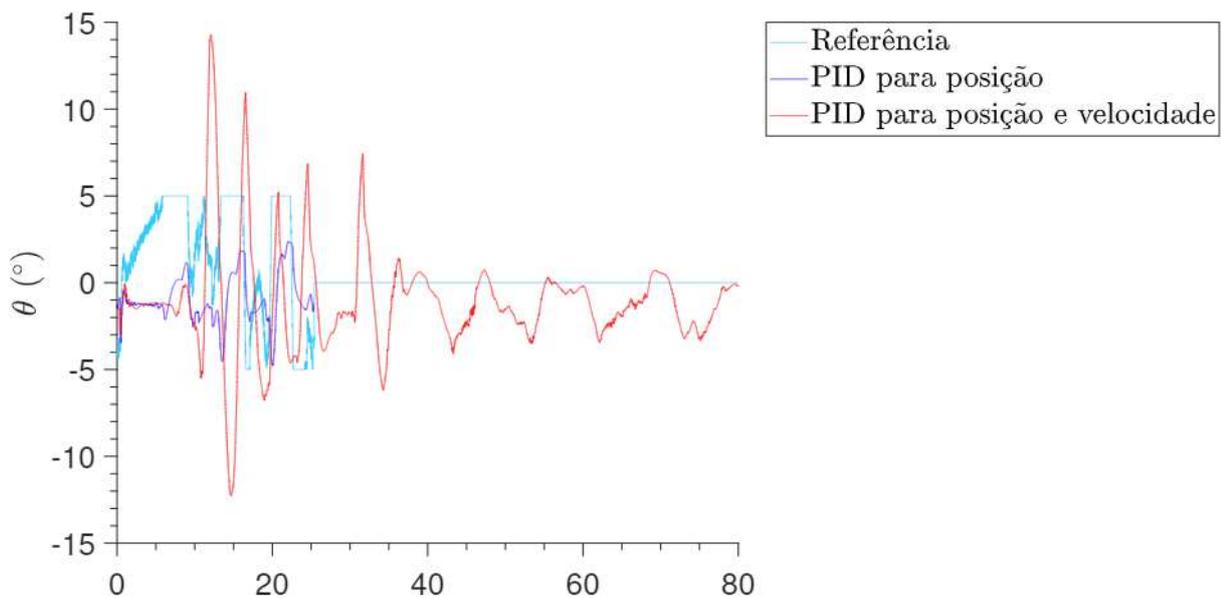
E, por fim, tem-se o gráfico do percurso que cada aeronave realizou. Para a comparação, este é um dos pontos mais importantes pois avalia-se a capacidade de cada estratégia de controle de cumprir uma missão, uma tarefa preestabelecida que, neste caso, é o trajeto. Como mostra a Figura 43, o controlador de posição concluiu o caminho proposto, passando pelos quatro pontos de checagem, enquanto a segunda estratégia passa apenas por dois e se perde

Figura 39 – Comportamento de cada estratégia de controle em relação ao valor de referência para o ângulo roll.



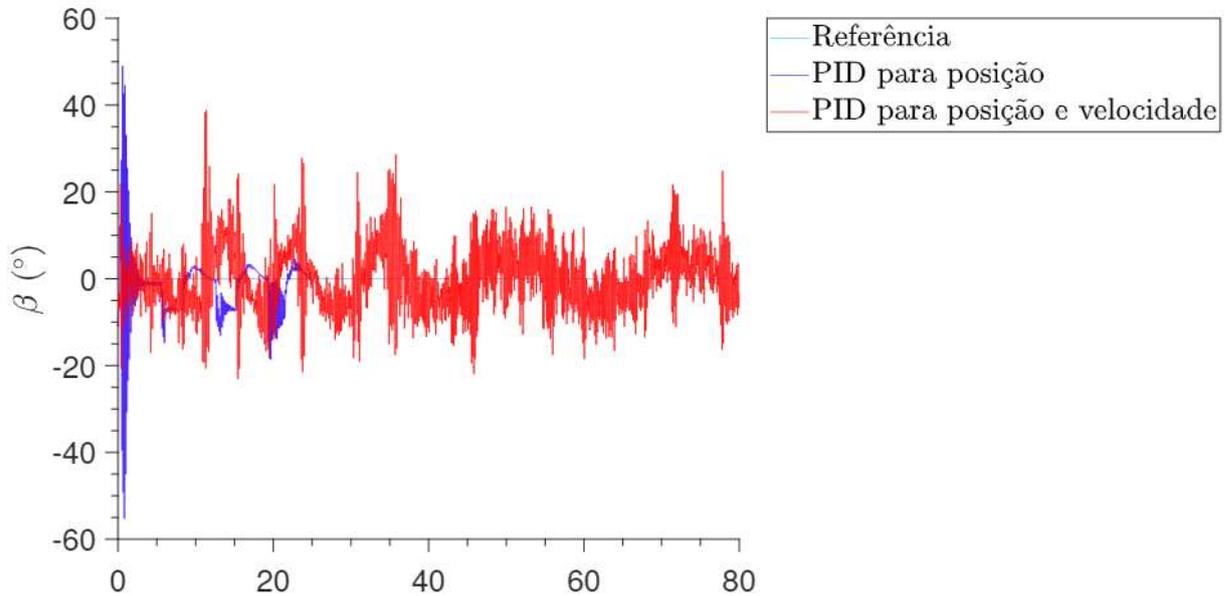
Fonte: Elaborada pela autora.

Figura 40 – Comportamento de cada estratégia de controle em relação ao valor de referência para o ângulo pitch.



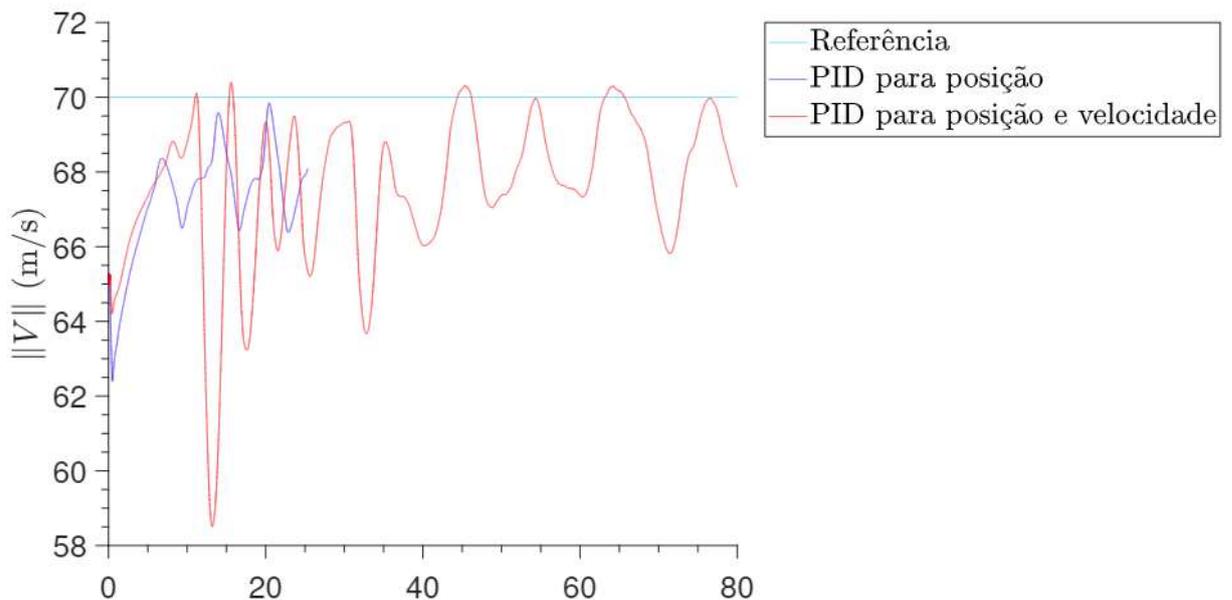
Fonte: Elaborada pela autora.

Figura 41 – Comportamento de cada estratégia de controle em relação ao valor de referência para o ângulo de sideslip.



Fonte: Elaborada pela autora.

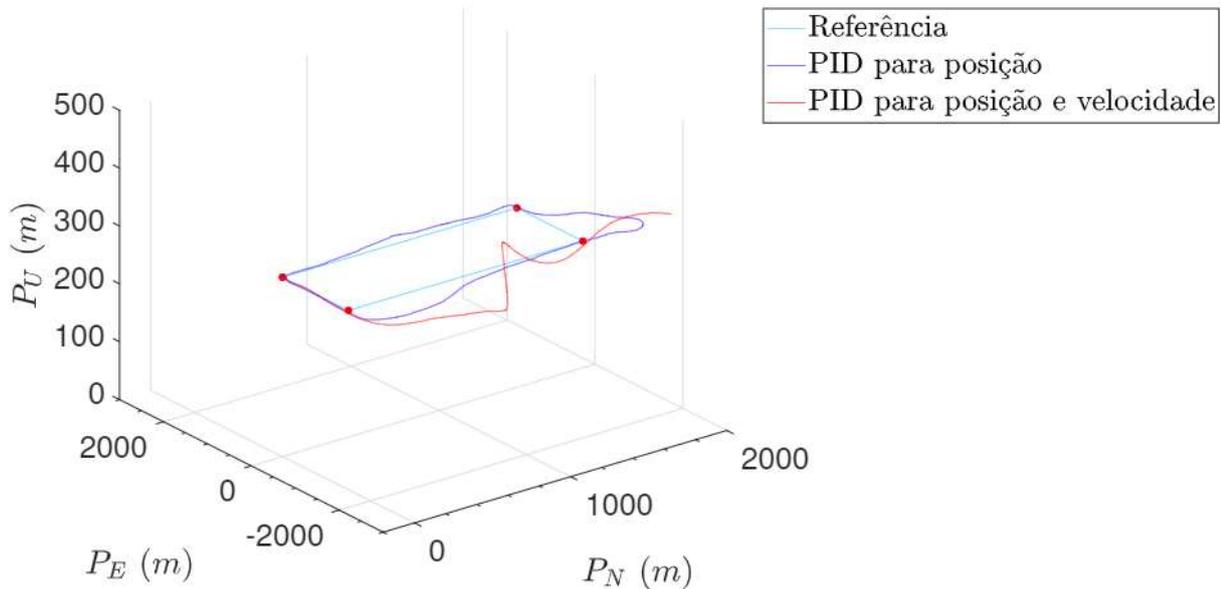
Figura 42 – Comportamento de cada estratégia de controle em relação ao valor de referência para a variável de velocidade.



Fonte: Elaborada pela autora.

em voo cruzeiro.

Figura 43 – Comportamento de cada conjunto de ganhos durante o seguimento de caminho.



Fonte: Elaborada pela autora.

Em detrimento ao Algoritmo Genético não executar seu melhor funcionamento e o ajuste de ganhos ter sido parcial ou insuficiente, nas condições apresentadas a estratégia de controle para posição mostrou-se eficiente ao cumprir a missão estabelecida. Embora o caminho trilhado não tenha sido exatamente o definido, a aeronave em seu controle passa pelos quatro *waypoints*, que definem o ponto em que o veículo deve fazer a manobra para mudar de posição.

Já a estratégia de controle para posição e velocidade não foi capaz de concluir o percurso e realizou apenas metade da missão atribuída. Este controlador possui uma abordagem mais complexa, que considera não só os ângulos, mas também as velocidades angulares do VANT. Isso pode implicar em uma necessidade maior da atuação dos controladores PID que, por sua vez, não propiciaram ganhos suficientes para a sustentação necessária neste caso.

5 CONCLUSÕES

Os experimentos de simulação e execução dos testes de *Software in the Loop* expressam a importância e eficiência desses métodos para obter resultados significativos na avaliação de controladores. Embora o ajuste de ganhos não tenha sido completamente bem sucedido, visto que o AG apresentou um comportamento inesperado devido a instabilidade na comunicação entre o simulador e o MATLAB, a aplicação de estratégias de otimização permitiu identificar imprecisões na tentativa de ajustar os ganhos para os controladores. Esses experimentos, combinados à análise estatística, permitiram encontrar o melhor conjunto de ganhos para os controladores PID aqui abordados, nas condições atingidas. Tendo cada estratégia em seu melhor desempenho alcançado dentro do escopo do projeto, foi possível compará-las de maneira justa e avaliar qual apresenta a melhor performance para Veículos Aéreos Não Tripulados de asa fixa.

A escolha do software de simulação, o X-Plane 11, também foi de suma importância para executar os testes mais próximos da realidade. Dessa forma é possível ter uma visão mais palpável sobre o comportamento da aeronave em condições de voo semelhantes às reais. Dito isso, destaca-se também a relevância de uma boa comunicação entre o simulador e o programa onde foram implementados e executados os controladores, o MATLAB.

Uma comunicação bidirecional fluida deve ser capaz de realizar a troca de informações entre os programas em tempo hábil de execução, de forma que os valores para as superfícies de controle cheguem corretamente, sem atrasos, e a simulação não seja afetada. Entretanto, durante a obtenção dos conjuntos de ganhos por meio do Algoritmo Genético, foi notado que o comportamento do algoritmo não era como o definido pela literatura e descrição da função que o executa. Iniciou-se, então, uma investigação para identificar as possíveis causas para o problema. Os motivos mais prováveis, em termos de código, foram mitigados por boas práticas durante a implementação dos controles e funções de simulação. Portanto, o comportamento presenciado é proveniente de falhas na comunicação entre o X-Plane 11 e o MATLAB, porém não há como conter ou sequer prever a perda de informações na comunicação pelo UDP, visto que a falta de confirmação e checagem de envio das mensagens é uma característica nata do protocolo.

Sendo o UDP a forma disponibilizada pelo X-Plane 11 para a comunicação com programas externos, não há como corrigir ou amenizar a perda de informações. Ainda assim, como visualizado pelos valores e gráficos das funções de custo, estima-se que a perda de informações não foi tão significativa, pois mesmo com esse ocorrido os ganhos obtidos pelo AG destacaram-se pelo desempenho e foram os melhores alcançados para as duas estratégias.

Para a comparação entre os controladores, um dos pontos principais foi tornar cada execução da simulação iguais. Para uma comparação precisa, é necessário que ambas tenham exatamente os mesmos recursos e condições, diferindo apenas no que é o alvo de confronto: os

controles PID. Para a estratégia de controle de posição, são considerados apenas as variáveis de controle dos ângulos *roll*, *pitch*, *sideslip* e a velocidade. Na estratégia de controle de posição e velocidade, em adição a estes, são consideradas também as velocidades angulares referentes à *roll* e *pitch*, expressadas por P e Q, respectivamente.

Visto que a maior atuação dos VANTs são pequenas missões, onde são preestabelecidos um trajeto e ações, o ponto de maior interesse ao comparar os controladores foi avaliar a capacidade de cada um conduzir a aeronave pelo caminho desejado. Mas todo o experimento realizado neste trabalho não envolveu apenas a comparação, mas também o ajuste de ganhos para os controladores PID que englobou uma pequena parte de uma grande área da computação, a otimização por meio de Algoritmos Genéticos.

Os experimentos mostraram que mesmo uma otimização consolidada pela literatura pode apresentar comportamentos divergentes por inúmeros fatores que não cabem à teoria, apenas à prática. Visto que o ajuste dos ganhos, uma parte indispensável para este trabalho, foi afetada pela comunicação entre o simulador e o MATLAB (responsável por executar toda a parte lógica), o desempenho dos controladores podem não ter sido maximizados.

Ainda assim, a estratégia de controle para posição, que possui uma abordagem menos completa à sua comparável, conseguiu concluir a missão definida. Isso implica que não necessariamente existem estratégias ruins ou sequer ideais, existem situações e inúmeros fatores que influenciam diretamente em seu desempenho. Algumas se enquadram melhor em determinadas situações e, neste caso, o PID para posição mostrou-se adaptável à situações adversas, onde os valores de seus ganhos não foram completamente satisfatórios, mas seu desempenho em geral sim.

5.1 Trabalhos futuros

Diante às limitações e lacunas encontradas durante a implementação e experimentos deste trabalho, é sugerido o aprofundamento no ajuste de ganhos dos controladores. Utilizando o Algoritmo Genético, recomenda-se como melhoria a regulação dos parâmetros utilizados, como o aumento da população e do número de gerações. Uma otimização mais longa, com maior número de parâmetros, pode aprimorar os resultados pois o AG é evolutivo e, portanto, com maior número de antepassados tende-se a encontrar melhores resultados finais. Entretanto, visto os problemas relacionados à comunicação entre o X-Plane 11 e o MATLAB, é interessante avaliar novas maneiras para obter esses ganhos. Existem inúmeros algoritmos de otimização, como a evolução diferencial, o *Sequential Quadratic Programming* (SQP), entre outros, que oferecem diferentes abordagens para otimizar os valores dos ganhos Proporcional, Integral e Derivativo do controlador.

Outro viés para melhorar a atuação das estratégias de controle é baseado em diminuir o efeito de *chattering* que foi identificado durante os experimentos, uma vez que a retirada do ganho K_D não foi suficiente para extingui-lo. Neste caso, é pertinente estudar a inclusão de filtros na ação do termo derivativo e a inclusão do esforço dos atuadores na função custo da

otimização, para que seja levado em consideração e, eventualmente, reduzir esse efeito.

Por fim, o fato do trabalho envolver a execução do MATLAB e do X-Plane na mesma máquina pode ter se tornado também um limitador de desempenho, visto que ambos os programas necessitam de uma alta atuação do processador central. Sendo assim, a melhoria do processamento através de *hardware* ou até mesmo a computação distribuída são boas alternativas para melhorar o tempo de processamento e qualidade da execução dos programas. Isso pode trazer melhores resultados e permite, também, a execução de novas técnicas para o ajuste e teste dos ganhos e estratégias.

Referências

- ÅSTRÖM, K.; HÄGGLUND, T. **Advanced Pid Control**. USA: ISA-The Instrumentation, Systems and Automation Society, 2006. 460 p. ISBN 9781556179426. Citado na página 8.
- ATLAS, E. et al. Hardware-in-the-loop test platform design for uav applications. In: TRANS TECH PUBL. **Applied Mechanics and Materials**. [S.l.], 2015. v. 789, p. 681–687. Citado na página 12.
- AUSTIN, R. **Unmanned Aircraft Systems: UAVS Design, Development and Deployment**. [S.l.]: Wiley, 2010. (Aerospace Series). ISBN 9780470664803. Citado 2 vezes nas páginas 1 e 5.
- BEARD, R. W.; MCLAIN, T. W. **Small unmanned aircraft: Theory and practice**. [S.l.]: Princeton university press, 2012. Citado na página 1.
- Bittar, A. et al. Guidance software-in-the-loop simulation using x-plane and simulink for uavs. In: **2014 International Conference on Unmanned Aircraft Systems (ICUAS)**. [S.l.: s.n.], 2014. p. 993–1002. Citado na página 12.
- CRAIGHEAD, J. et al. A survey of commercial open source unmanned vehicle simulators. In: **2007 IEEE International Conference on Robotics and Automation**. [S.l.: s.n.], 2007. p. 852–857. Citado 2 vezes nas páginas 1 e 15.
- Ito, F. S. C. et al. Cooperative uav formation control simulated in x-plane. In: **2017 International Conference on Unmanned Aircraft Systems (ICUAS)**. [S.l.: s.n.], 2017. p. 1522–1529. Citado na página 12.
- JAIN, R. **The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling**. Wiley, 1990. ISBN 9788126519057. Disponível em: <<https://books.google.com.br/books?id=eOR0kJgMqkC>>. Citado na página 29.
- JOHNSON, E.; FONTAINE, S. Use of flight simulation to complement flight testing of low-cost uavs. In: **AIAA Modeling and Simulation Technologies Conference and Exhibit**. [S.l.: s.n.], 2001. ISBN 9781563479458. Citado na página 11.
- KIMATHI, S. Application of reinforcement learning in heading control of a fixed wing uav using x-plane platform. IJSTR, 2017. Citado na página 11.
- KNOSPE, C. Pid control. **IEEE Control Systems Magazine**, IEEE, v. 26, n. 1, p. 30–31, 2006. Citado na página 31.
- LAMINAR RESEARCH. **New in X-Plane 11**. 2016. Disponível em: <<https://www.x-plane.com/>>. Acesso em: 11 out. 2019. Citado na página 15.
- MATHWORKS. **MATLAB**. 2019. Disponível em: <https://www.mathworks.com/products/matlab.html?s_tid=hp_ff_p_matlab>. Acesso em: 11 out. 2019. Citado na página 15.
- MATHWORKS. **How the Genetic Algorithm Works**. 2020. Disponível em: <<https://www.mathworks.com/help/gads/how-the-genetic-algorithm-works.html>>. Acesso em: 28 ago. 2020. Citado 2 vezes nas páginas 22 e 38.

Michailidis, M. G. et al. A software in the loop (sil) kalman and complementary filter implementation on x-plane for uavs. In: **2019 International Conference on Unmanned Aircraft Systems (ICUAS)**. [S.l.: s.n.], 2019. p. 1069–1076. Citado na página 11.

MITCHELL, M. **An Introduction to Genetic Algorithms**. Cambridge, MA, USA: MIT Press, 1998. ISBN 0262631857. Citado 3 vezes nas páginas 11, 22 e 38.

NATIONAL IMAGERY AND MAPPING AGENCY. **Department of Defense World Geodetic System 1984: its definition and relationships with local geodetic systems**. 3rd. ed. St. Louis, MO, USA, 2000. Accessed: 27/02/2018. Disponível em: <http://earth-info.nga.mil/GandG/publications/tr8350.2/tr8350_2.html>. Citado na página 19.

NISE, N. S. **Engenharia de Sistemas de Controle**. [S.l.]: LTC, 2012. v. 6. ISBN 978-0470-54756-4. Citado na página 8.

OGATA, K. **Modern Control Engineering**. [S.l.]: Prentice Hall, 2010. (Instrumentation and controls series). ISBN 9780136156734. Citado 3 vezes nas páginas 1, 7 e 8.

PELIZER, G. V.; SILVA, N. B. F.; BRANCO, K. R. L. J. C. 3D path-following algorithms for unmanned aerial vehicles adjusted with genetic algorithm. In: BRANCO, K.; PINTO, A.; PIGATTO, D. (Ed.). **Communication in Critical Embedded Systems: WoCCES 2013, WoCCES 2014, WoCCES 2015, WoCCES 2016, Revised Selected Papers**. Brazil: Springer International Publishing, 2017. p. 63–80. Citado na página 19.

RAHIMI, M. R.; HAJIGHASEMI, S.; SANAEI, D. Designing and simulation for vertical moving control of uav system using pid, lqr and fuzzy logic. **International Journal of Electrical and Computer Engineering**, Citeseer, v. 3, n. 5, p. 651, 2013. Citado na página 13.

SILVA, N. B. F. et al. Development of a fixed-wing vertical takeoff and landing aircraft as an autonomous vehicle. In: **2017 Latin American Robotics Symposium (LARS) and 2017 Brazilian Symposium on Robotics (SBR)**. Curitiba, Brazil: [s.n.], 2017. p. 1–6. Citado na página 12.

SILVA, N. B. F. et al. Dynamic inversion and gain-scheduling control for an autonomous aerial vehicle with multiple flight stages. **Journal of Control, Automation and Electrical Systems**, v. 29, n. 3, p. 328–339, Jun 2018. ISSN 2195-3899. Citado 3 vezes nas páginas 5, 6 e 7.

SILVA, N. B. F. da. **Development of an autonomous unmanned aerial vehicle specification of a fixed-wing vertical takeoff and landing aircraft**. Tese (Doutorado) — Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP, Brasil, 2018. Citado na página 5.

TANENBAUM, A. S. **Redes de Computadores**. trad. 4 ed. Rio de Janeiro: Elsevier, 2003. Citado na página 38.

VALAVANIS, K. P. **Advances in Unmanned Aerial Vehicles: State of the Art and the Road to Autonomy**. 1st. ed. USA: Springer Publishing Company, Incorporated, 2007. ISBN 1402061137, 9781402061134. Citado na página 1.

VALAVANIS, K. P.; VACHTSEVANOS, G. J. **Handbook of Unmanned Aerial Vehicles**. [S.l.]: Springer Publishing Company, Incorporated, 2015. ISBN 9048197082, 9789048197088. Citado 2 vezes nas páginas 1 e 5.

ZHANG, C.; KOVACS, J. The application of small unmanned aerial systems for precision agriculture: a review. **Precision Agriculture**, Springer US, v. 13, n. 6, p. 693–712, 2012. ISSN 1385-2256. Disponível em: <<http://dx.doi.org/10.1007/s11119-012-9274-5>>. Citado na página 1.