Engenharia Mecatrônica
Implementação do *Q-learning* no rastreamento de referências constantes em um aeropêndulo

Cleiton Kennedy de Morais Filho

Universidade Federal de Uberlândia
Faculdade de Engenharia Mecânica
Engenharia Mecatrônica

Uberlândia
2023

Cleiton Kennedy de Morais Filho

Engenharia Mecatrônica
Implementação do *Q-learning* no rastreamento de referências constantes em um aeropêndulo

Trabalho de Conclusão de Curso apresentado à Faculdade de Engenharia Mecânica da Universidade Federal de Uberlândia como requisito parcial para obtenção do título de Bacharel em Engenharia Mecatrônica.

Área de concentração: Engenharia Mecatrônica

Orientador: Pedro Augusto Queiroz de Assis

Uberlândia
2023

Agradecimentos

Aos meus pais, pelo amor e incentivo incondicional neste longo processo.

À minha digníssima esposa e ao querido Bayard, pelo acalento nos dias difíceis.

Às minhas irmãs, por serem um exemplo de perseverança e criatividade.

Ao professor Pedro Assis, por todo apoio e paciência sem os quais eu não teria conseguido.

"Quebre o padrão que liga os itens do aprendizado e você necessariamente destrói toda a qualidade."

(Gregory Bateson)

Resumo

O objetivo deste trabalho é aplicar um método de aprendizado de máquina no controle de um aeropêndulo. Especificamente, o método *Q-Learning* de aprendizado por reforço é utilizado para fazer com que a haste do aeropêndulo seja guiada para uma referência. Para isso, primeiramente um modelo matemático não linear para descrever a dinâmica do sistema é desenvolvido. Mais ainda, o espaço de estados e ações admissíveis foram discretizados, e foi definida uma função de recompensas para fazer com que o agente aprenda a realizar a tarefa de controle. Tal função retorna recompensas positivas ao agente caso a haste permaneça em um ângulo de referência com a base. Já recompensas negativas são atribuídas quando um estado terminal é atingido. As ações possíveis de serem tomadas em um estado são acelerar ou desacelerar o motor. A magnitude da aceleração/desaceleração depende da magnitude do erro de rastreamento. Análises do número de episódios de treinamento necessários para que o agente aprenda a realizar a tarefa, e da influência dos parâmetros de ajuste no aprendizado são realizadas. Os resultados mostram que o agente aprendeu a controlar o aeropêndulo sem erro de rastreamento em regime permanente em todos os casos após o treinamento, indicando a viabilidade da aplicação deste tipo de método no controle de sistemas não lineares. Identificou-se que são necessários aproximadamente 2900 episódios para que o aprendizado seja concluído. Mais ainda, verificou-se que ao aumentar o fator de ganância e a taxa de aprendizagem, a resposta transitória do sistema melhorou. Já variando-se o fator de desconto, os melhores resultados foram obtidos com valores intermediários.

Palavras-chave: Aprendizado por reforço; *Q-learning*; Aeropêndulo; Controle em malha fechada.

Lista de ilustrações

Figura 1 – Protótipo do aeropêndulo. 12
Figura 2 – Vista frontal do protótipo do aeropêndulo. 13
Figura 3 – *encoder* modelo Omron E6B2. 14
Figura 4 – Controlador eletrônico de velocidade. 14
Figura 5 – Hélice do conjunto propulsor. 15
Figura 6 – Motor sem escovas. 16
Figura 7 – Controlador STM32F103. 16
Figura 8 – Diagrama de blocos ilustrando o funcionamento do sistema. 17
Figura 9 – Diagrama de forças do aeropêndulo. 18
Figura 10 – Diagrama de interação entre agente e ambiente em um problema de aprendizado por reforço (SUTTON; BARTO, 2018). 22
Figura 11 – Ilustração do funcionamento da política ϵ-*greedy*. 25
Figura 12 – Diagrama *Q-learning* utilizando a política ϵ-*greedy*. 26
Figura 13 – Diagrama simplificado da arquitetura da solução proposta. 30
Figura 14 – Exemplo de interação com o programa via terminal. 30
Figura 15 – Análise do impacto do número de episódios no rastreamento da referência. 31
Figura 16 – Refinamento da análise do número de episódios para treinamento do agente. 32
Figura 17 – Desempenho do agente no último episodio de treinamento para diferentes valores de ε. 34
Figura 18 – Desempenho do agente no último episodio de treinamento para diferentes valores de γ. 35
Figura 19 – Desempenho do agente no último episodio de treinamento para diferentes valores de α. 35
Figura 20 – Desempenho do agente utilizando os parâmetros definidos após o refinamento. 36

Figura 21 – Comparação da resposta do sistema após o treinamento utilizando os incrementos de velocidade de rotação iniciais (Tabela 3) e após o refinamento (Tabela 4). 37
Figura 22 – Simulação após o treinamento para a referência de 5°. 38
Figura 23 – Simulação após o treinamento para a referência de 10°. 38
Figura 24 – Simulação após o treinamento para a referência de 15°. 39
Figura 25 – Simulação após o treinamento para a referência de 20°. 39
Figura 26 – Simulação após o treinamento para a referência de 25°. 40

Lista de tabelas

Tabela 1 – Especificações técnicas STM32F103 (STMICROELECTRONICS, 2022). 17
Tabela 2 – Descrição dos símbolos usados no modelo. 18
Tabela 3 – Valores dos incrementos aplicados a ω^2 de acordo com o módulo do erro de rastreamento. 28
Tabela 4 – Valores dos incrementos aplicados a ω^2 de acordo com o erro de rastreamento. 36

Sumário

1 INTRODUÇÃO 9
2 DESCRIÇÃO DO SISTEMA 12
2.1 Sensor de posição 13
2.2 Conjunto propulsivo 14
2.2.1 Hélice 15
2.2.2 Motor sem escovas 15
2.3 Arquitetura do sistema 16
2.4 Modelagem Matemática 17
3 APRENDIZADO POR REFORÇO 20
3.1 Aprendizado Supervisionado 21
3.2 Aprendizado Não Supervisionado 21
3.3 Aprendizado por Reforço 21
3.3.1 Estrutura do aprendizado baseado em reforço 21
3.3.2 Função Valor 23
3.3.3 Diferença Temporal 24
3.3.4 *Q-Learning* 24
4 RESULTADOS 27
4.1 Material utilizado 27
4.1.1 Hardware 27
4.1.2 Software 27
4.2 Arquitetura da simulação 28
4.3 Análise de resultados 31
4.3.1 Análise do número de episódios de treinamento 31
4.3.2 Análise da sensibilidade 32
4.3.3 Análise de resultados para diferentes referências 36
5 Conclusão 41
Referências 43

Capítulo 1

INTRODUÇÃO

Tipicamente uma das primeiras etapas no desenvolvimento de controles automáticos é a criação de um modelo matemático para o processo a ser controlado (DORF; BISHOP, 2001). Esse modelo pode ser composto por um conjunto de equações que representam aproximadamente a dinâmica entre entradas e saídas do processo real. Durante a modelagem é necessário buscar um equilíbrio entre o esforço envolvido no desenvolvimento do modelo, o nível de detalhes considerados e os benefícios esperados ao se utilizar tal representação. O processo de modelagem é sensível a erros matemáticos, o que pode prejudicar o projeto ou o desempenho do controlador (GARCIA, 2005).

De forma geral, os modelos matemáticos podem ser divididos em duas categorias: lineares ou não lineares. Os sistemas lineares obedecem às propriedades da homogeneidade e da aditividade (princípio da superposição). Essas propriedades tornam mais fácil a resolução de problemas complexos, porque permitem a obtenção do resultado final por meio da análise das respostas individuais de cada entrada, ou seja, é garantida uma relação linear entre a entrada e a saída. Entretanto, a maioria das representações matemáticas de sistemas físicos não são lineares, pois não obedecem aos princípios citados e, consequentemente, não podem ser analisadas de forma fracionada (OGATA, 2011).

Para controlar sistemas não lineares, as estratégias convencionais utilizam processos de linearização. Tais processos consistem na obtenção de sistemas lineares, utilizando-se, por exemplo, a série de Taylor. Essa abordagem envolve a expansão da função não linear que descreve o sistema em uma série de potências ao redor de um ponto de operação, de forma que possa ser considerado linear (OGATA, 2011).

Como alternativa, podem ser empregadas técnicas de controle que independem de modelos matemáticos, sendo essas aplicáveis a modelos lineares ou não. Uma dessas técnicas é o aprendizado de máquina, que consiste na utilização de métodos computacionais com o objetivo de produzir um sistema capaz de tomar decisões com base em experiências obtidas previamente. Os diferentes métodos de aprendizagem são definidos pela maneira com que o sistema obtém tal experiência e pela forma como isso é feito (MONARD; BARANAUSKAS, 2003). Um desses métodos é o aprendizado por reforço, o qual consiste

na definição de um agente, de um ambiente (com o qual o agente irá interagir) e de uma função de recompensas. Com estes elementos, constitui-se um processo de decisões em que reforços positivos ou negativos são aplicados ao agente de acordo com os resultados de suas ações no ambiente. Na prática, as recompensas são utilizadas para quantificar as ações tomadas, de forma que o conjunto de regras que guia a tomada de decisão (política) possa ser melhorado.

Dentre os tipos de aprendizado de reforço, duas classificações se destacam: os métodos de Monte Carlo e os métodos de aprendizado por Diferença Temporal (DT). Os métodos de Monte Carlo são desenvolvidos definindo-se uma função valor que considera a expectativa de retorno para atribuir um valor $V^{\pi}(s_t)$ para cada estado. Essa expectativa é obtida calculando-se a média das recompensas obtidas após um número finito de interações do agente com o ambiente. Em seguida, o agente utiliza uma política e aplica as ações de acordo com os valores atribuídos a todos os estados. Note que, nos métodos de Monte Carlo, primeiramente é necessário um número significativo de experiências para que os valores dos estados sejam estimados. No aprendizado por DT, também é definida a função valor, entretanto, o agente atualiza $V^{\pi}(s_t)$ empregando apenas uma parte da experiência que já foi adquirida. Para isso, utiliza-se a diferença entre esses valores ao longo de determinado intervalo de tempo. Em outras palavras, compara-se a diferença de valor entre os estados, tornando possível avaliar o impacto de uma ação sem a necessidade de concluir todo o processo de interação com o ambiente (SUTTON; BARTO, 2018).

Os métodos que utilizam DT se diferenciam, basicamente, pela maneira com que o ciclo de interação do agente com o ambiente ocorre. Contudo, ao invés de se adotar a função valor $V^{\pi}(s_t)$, define-se uma função valor de ação $Q^{\pi}(s_t,a_t)$, que basicamente representa o retorno esperado ao se seguir a ação a_t no estado s_t e, então, seguir a política π. Por exemplo, no algoritmo SARSA, esse ciclo se define por: Leitura do estado s_t, tomada de ação a_t, cálculo da recompensa r_t, leitura do estado seguinte s_{t+1}, escolha da ação seguinte a_{t+1}, atualização do valor $Q(s_t,a_t)$ de acordo com uma estimativa de $Q^{\pi}(s_{t+1},a_{t+1})$. Desta forma, o agente atualiza o valor de cada par estado-ação considerando principalmente a ação escolhida no estado seguinte a_{t+1}. Já no algoritmo *Q-learning*, também utiliza-se a função valor de ação $Q^{\pi}(s_t,a_t)$, mas o ciclo de interação do agente com o ambiente define-se por: Leitura do estado s_t, tomada de ação a_t, atribuição da recompensa r_t, leitura do estado seguinte s_{t+1}, atualização dos valores de $Q^{\pi}(s_t,a_t)$ baseando-se no valor máximo de $Q^{\pi}(s_{t+1},a)$. Portanto, enquanto o algoritmo SARSA depende da ação tomada no estado seguinte e, consequentemente, da política adotada, o *Q-learning* se baseia no valor máximo de Q do estado seguinte. Por isso, ao não depender da política adotada, é considerado um método *off-policy* (SUTTON; BARTO, 2018).

Neste contexto, como um primeiro contato do discente com a área de aprendizado de máquina, este trabalho envolve o emprego do *Q-learning* no controle de um aeropêndulo em ambiente de simulação. O aeropêndulo foi escolhido por não apresentar complexidade

elevada e abranger diversos parâmetros comuns a problemas tradicionais de controle, além de possuir um modelo não linear. Em suma, o aeropêndulo é composto por uma base, um eixo rotacional e uma haste. Além disso, possui um conjunto propulsor que ao ser acionado movimenta a haste em torno do eixo rotacional. Um projeto de controlador para o aeropêndulo deve ser capaz de fazer com que o ângulo entre a haste e a base seja mantido em um valor de referência especificado manipulando-se a velocidade de rotação do motor.

O restante do presente documento está estruturado da seguinte forma:

- No Capítulo 2 são descritos os componentes do aeropêndulo bem como sua arquitetura de funcionamento e modelagem matemática.
- O Capítulo 3 trata dos conceitos teóricos de aprendizado por reforço, contextualizando o tema e definindo o método que será utilizado nas simulações.
- No Capítulo 4 é apresentada a arquitetura da solução, o material utilizado, a estratégia adotada para realizar as simulações e a análise dos resultados. Tais resultados foram obtidos realizando-se simulações de treinamento e calibração, e incluem gráficos de ângulo e velocidade angular que buscam validar os objetivos específicos.
- O Capítulo 5 apresenta a conclusão do trabalho, cita os resultados obtidos e propõe sugestões para pesquisas futuras.

Capítulo 2

DESCRIÇÃO DO SISTEMA

O objetivo do presente trabalho é empregar uma técnica de aprendizado por reforço no controle de um aeropêndulo. Para isso, será utilizada uma simulação computacional do sistema. Essa abordagem foi escolhida porque permite que a avaliação de desempenho e a correção de possíveis erros ocorram rapidamente. Entretanto, para tornar o entendimento mais claro, a seguir detalham-se as características construtivas de um protótipo do aeropêndulo.

A estrutura do aeropêndulo é similar a de um pêndulo invertido: têm-se base, eixo rotacional e haste. Na extremidade desta última, é posicionado um motor sem escovas no qual é acoplada uma hélice. Consequentemente, quando o motor é acionado, a força de empuxo resultante causa um movimento da haste em torno do eixo de rotação. As Figuras 1 e 2 mostram fotos do protótipo completo.

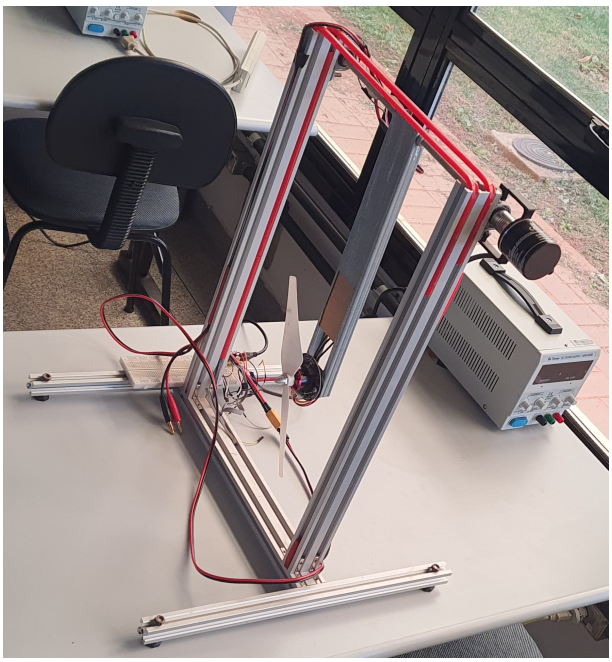

Figura 1 – Protótipo do aeropêndulo.

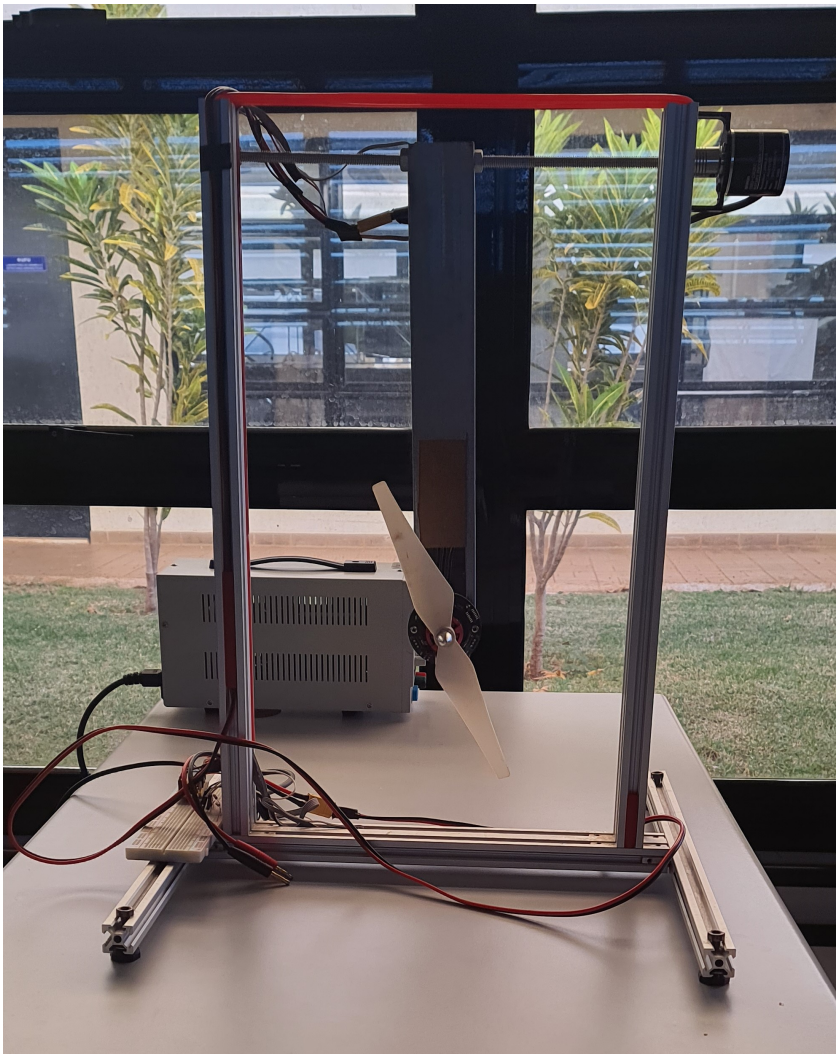

Figura 2 – Vista frontal do protótipo do aeropêndulo.

A estrutura é feita utilizando perfis de alumínio. No eixo de rotação foi inserido um rolamento para que o atrito da haste ao rotacionar seja reduzido.

O controle da velocidade de rotação do motor sem escovas é realizado por um controlador eletrônico de velocidade ESC (*Electronic Speed Controler*). O deslocamento da haste é medido por um *encoder*. Estes componentes são detalhados a seguir.

2.1 Sensor de posição

Para obter a posição da haste em relação ao eixo vertical é utilizado um *encoder* incremental. Especificamente, foi escolhido o *encoder* do modelo Omron E6B2, ilustrado na Figura 3. Com uma contagem de 2000 pulsos por revolução, esse *encoder* apresenta uma precisão de aproximadamente 0,18°.

Figura 3 – *encoder* modelo Omron E6B2.

2.2 Conjunto propulsivo

O sistema de atuação é composto por um motor sem escovas, um ESC e uma hélice. A velocidade angular do motor é mensurada por meio de um sensor de efeito Hall.

O ESC recebe um comando fornecido pelo controlador, que é embarcado em um STM32, e gera sinais do tipo PWM (*Pulse Width Modulation*), criado através da comutação dos MOSFETs, que são impostos ao motor. A ideia é fazer com que o campo magnético do enrolamento do motor seja alterado no momento apropriado para aumentar ou diminuir a velocidade de rotação do motor (NEDELKOVSKI, 2021).

Para a fabricação do aeropêndulo, foi utilizado um controlador ESC da marca Emax capaz de suportar uma corrente contínua de 30A o qual é mostrado na Figura 4.

Figura 4 – Controlador eletrônico de velocidade.

Como já mencionado, o ESC é utilizado para controlar a velocidade de giro de um motor sem escovas e entre as vantagens de sua utilização podem ser citadas: boa eficiência energética, possibilidade de programar parâmetros de operação e proteção contra tensão baixa, perda de sinal e temperatura elevada (NAGEL, 2022).

2.2.1 Hélice

A Figura 5 mostra a hélice utilizada, trata-se do modelo 1045 de aproximadamente 25,4 cm de diâmetro e 1 mm de espessura.

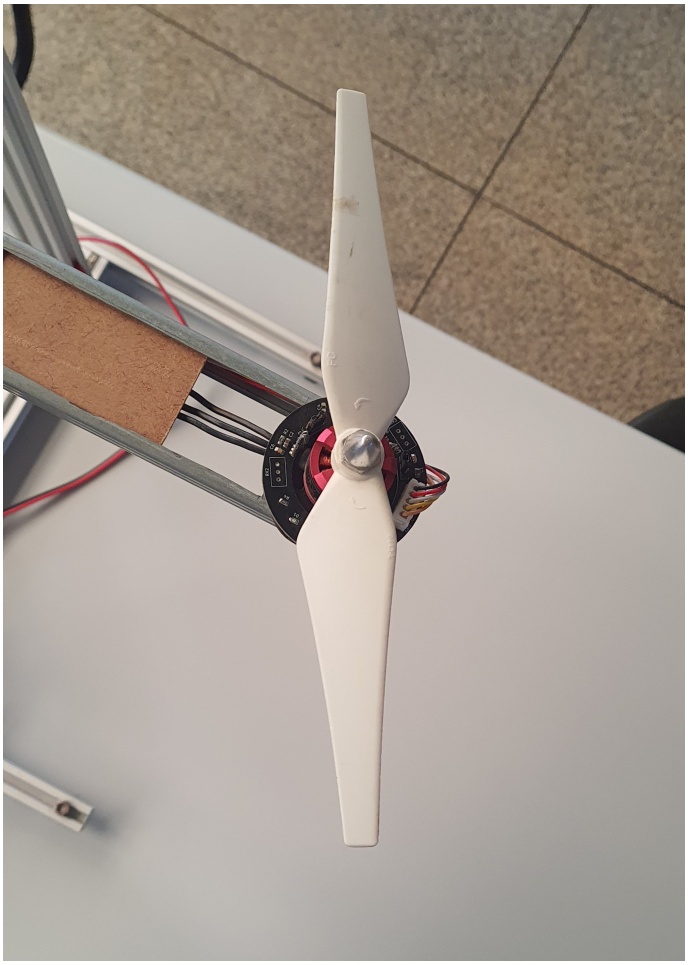

Figura 5 – Hélice do conjunto propulsor.

2.2.2 Motor sem escovas

O funcionamento de um motor sem escovas ocorre através da comutação sequencial dos diferentes enrolamentos do estator, de forma que, o torque gerado pelo campo magnético movimente o rotor.

É possível controlar o torque gerado no motor e, consequentemente, a velocidade de rotação, manipulando-se a velocidade de comutação dos enrolamentos. Para isto emprega-se o ESC.

A Figura 6 mostra o motor compondo o conjunto propulsivo. Foi utilizado um modelo 2212 da marca Dji. Este modelo atua com uma taxa de $920\frac{rpm}{V}$ e a tensão admissível varia entre 7,4V e 14,8V, portanto esse motor pode operar entre 6808 rpm e 13616 rpm.

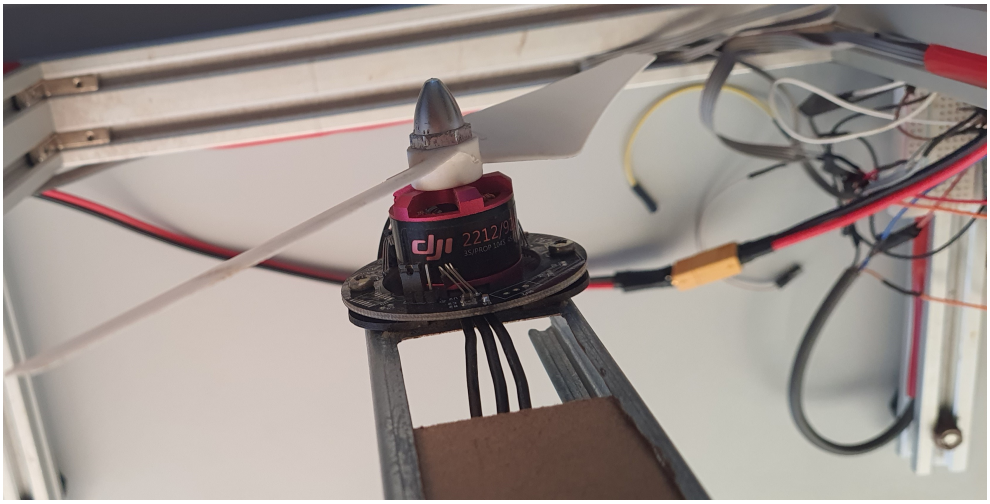

Figura 6 – Motor sem escovas.

2.3 Arquitetura do sistema

Para comunicação com os componentes eletrônicos do aeropêndulo, é adotado um microcontrolador da família STM32F103 mostrado na Figura 7. Este componente é equipado com núcleo ARM Cortex-M3 e arquitetura 32 bits, o que o permite uma abrangência de possibilidades de aplicação (BROWN, 2016).

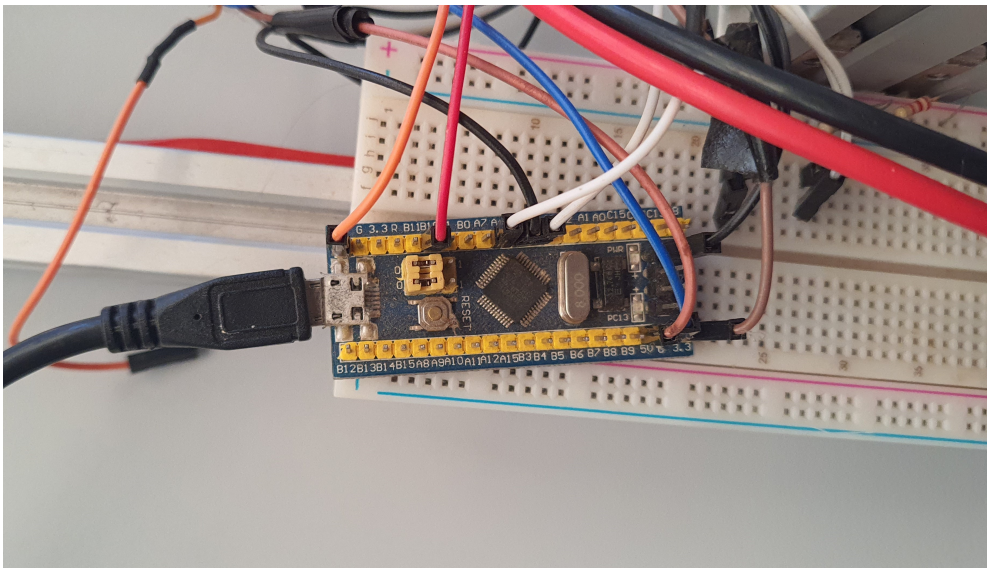

Figura 7 – Controlador STM32F103.

A Tabela 1 apresenta as especificações técnicas do microcontrolador usado.

Tabela 1 – Especificações técnicas STM32F103 (STMICROELECTRONICS, 2022).

Arquitetura	ARM Cortex-M3 32-bit RISC
Flash Memory	64 Kb
Clock	72 MHz
Interfaces	USARTs, SPIs, I2Cs, USB, CAN
SRAM	20Kb
Tensão	1,8 - 3,6V

O microcontrolador será responsável por receber os dados do sensor hall, do *encoder* e do ESC, e enviá-los ao computador, quando requisitado. O computador irá processar os dados e calcular a ação de controle (i.e. a velocidade de rotação) de acordo com o algoritmo de aprendizado por reforço desenvolvido.

Após calculada a velocidade de rotação, o computador enviará o comando para o microcontrolador que irá acionar o ESC para que se imponha ao motor a velocidade de giro desejada. Esta arquitetura está ilustrada no diagrama da Figura 8.

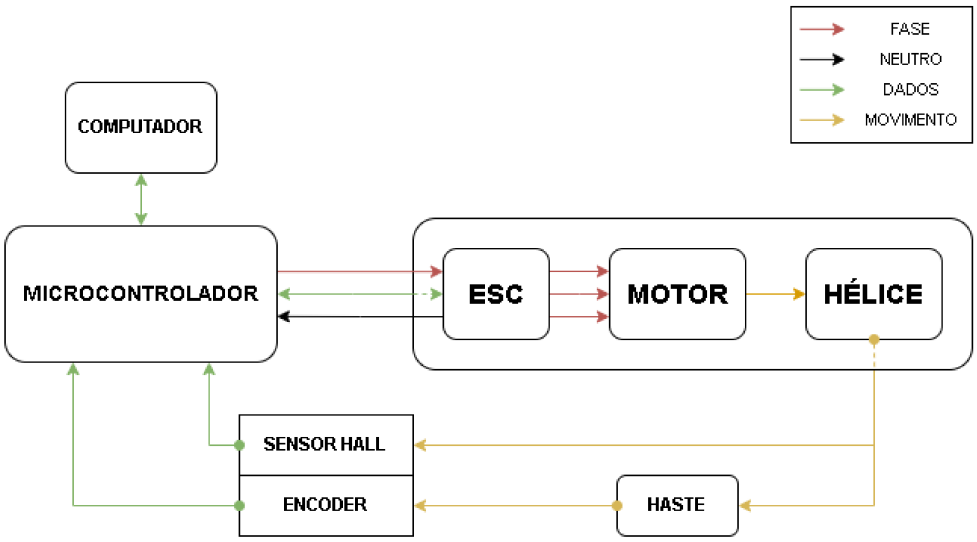

Figura 8 – Diagrama de blocos ilustrando o funcionamento do sistema.

2.4 Modelagem Matemática

Um corpo rígido é aquele em que a distância entre dois pontos quaisquer do corpo não varia. Para realização da modelagem da dinâmica do aeropêndulo, considera-se o conjunto propulsivo um corpo rígido e desconsidera-se a massa da haste. Dessa forma, têm-se atuando no corpo as forças e torques ilustrados na Figura 9. Nesta figura, F_e representa a força de empuxo gerada pelo conjunto propulsor, P é a força peso do corpo rígido e T_a representa um torque de atrito viscoso no rolamento. Mais ainda, L é o comprimento da haste, ω é a velocidade de rotação do motor e g é a aceleração gravitacional.

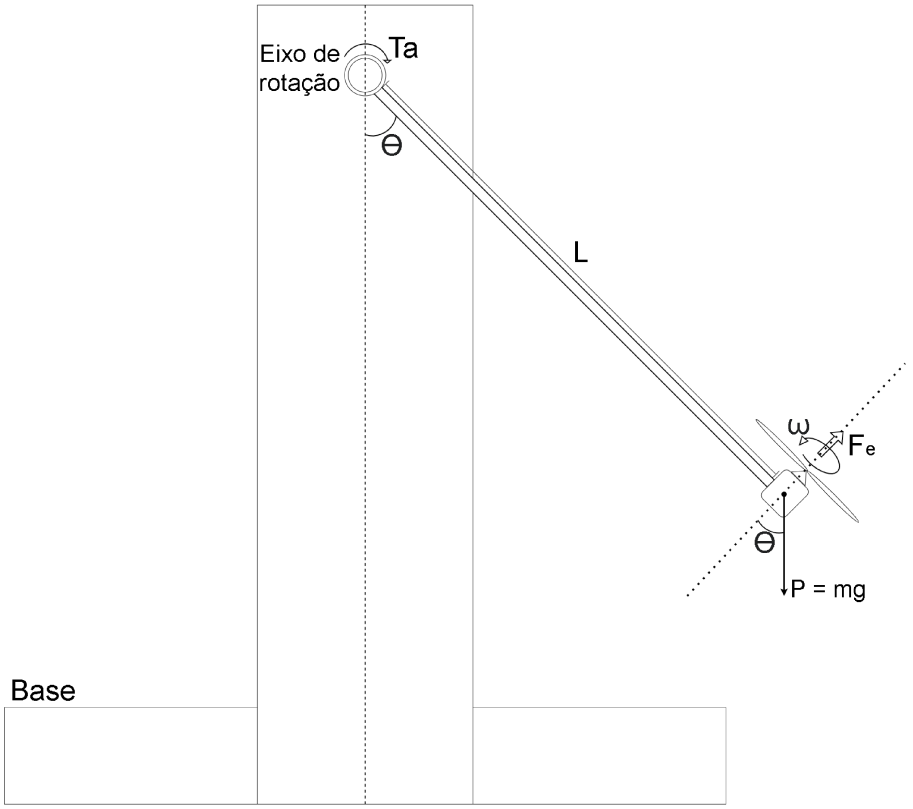

Figura 9 – Diagrama de forças do aeropêndulo.

A descrição e os valores das constantes do modelo matemático são detalhadas na Tabela 2.

Tabela 2 – Descrição dos símbolos usados no modelo.

Símbolo	Descrição	Valor
m	Massa do sistema	0,3182 kg
g	Aceleração da gravidade	9,81 m/s^2
b	Coeficiente de atrito viscoso	0,006856
L	Comprimento da haste	0,32 m
K_h	Constante de empuxo da hélice	$2{,}1 \times 10^{-5}$ Nm/V
I	Momento de inércia	0,0264 kgm^2
F_e	Força exercida pelo torque gerado pelo motor	-
ω	Velocidade de rotação do motor	-
θ	Ângulo formado entre a haste e o eixo de rotação	-
$\dot{\theta}$	Velocidade angular da haste	-
$\ddot{\theta}$	Aceleração angular da haste	-

Analisando a Figura 3, utiliza-se o eixo de simetria para aplicar a segunda lei de Newton para o movimento rotacional em torno do eixo de rotação. Para isto, definem-se os momentos realizados pela componente da força peso M_p, pela força de atrito viscoso M_{at} e pela inércia M_{ine}, representados respectivamente, por

$$M_p(t) = mg\,\mathrm{sen}(\theta(t))L \tag{1}$$

$$M_{at}(t) = b\dot{\theta}(t) \tag{2}$$

$$M_{ine}(t) = I\ddot{\theta}(t) \tag{3}$$

Assim, o momento realizado pela força de empuxo F_e aplicada na haste é igualado com os momentos previamente definidos. Matematicamente,

$$M_{F_e}(t) = M_p(t) + M_{at}(t) + M_{ine}(t) \tag{4}$$

Substituindo (1), (2) e (3) em (4), obtém-se

$$F_e(t)L = mg\,\mathrm{sen}(\theta(t))L + b\dot{\theta}(t) + I\ddot{\theta}(t) \tag{5}$$

Além disso, assume-se que a força de empuxo se relaciona com a velocidade de rotação do motor do seguinte modo:

$$F_e(t) = K_h\omega^2(t) \tag{6}$$

Substituindo (6) em (5) e isolando $\ddot{\theta}(t)$ é possível obter

$$\ddot{\theta}(t) = \frac{LK_h}{I}\omega^2(t) - \frac{mgL}{I}\,\mathrm{sen}(\theta(t)) - \frac{b}{I}\dot{\theta}(t) \tag{7}$$

Esse sistema pode ser representado no espaço de estados. Para isso, define-se o seguinte vetor de estados:

$$\mathbf{X}(t) = \begin{bmatrix} X_1(t) \\ X_2(t) \end{bmatrix} = \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \end{bmatrix} \tag{8}$$

Então, pode-se escrever o modelo final do aeropêndulo

$$\begin{bmatrix} \dot{X}_1(t) \\ \dot{X}_2(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{mgL}{I}\,\mathrm{sen}(\theta(t)) & -\frac{b}{I} \end{bmatrix} \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{LK_h}{I} \end{bmatrix} \omega^2(t) \tag{9}$$

$$\mathbf{Y}(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \end{bmatrix} \tag{10}$$

O modelo formado por (9) e (10) foi usado nas simulações descritas no Capítulo 4.

Capítulo 3

APRENDIZADO POR REFORÇO

O aprendizado de máquina se refere ao desenvolvimento de programas de computador que sejam capazes de melhorar seu desempenho por meio de experiências. Mais especificamente, encontra-se a seguinte definição na literatura:

> Diz-se que um programa de computador aprende com a experiência E em relação a alguma classe de tarefas T e a medida de desempenho P, se seu desempenho em tarefas de T, medido por P, melhora com a experiência E.
> -(MITCHELL, 1997)

Para representar o problema de controlar o aeropêndulo como na tarefa de aprendizado de máquina, pode-se definir tarefa, índice de desempenho e experiência como apontado a seguir:

- Tarefa T: Manter o aeropêndulo em uma posição desejada.
- Índice de desempenho P:
  - Erro entre o ângulo de referência e o ângulo de convergência da haste.
  - Os valores da tabela de *Q-values*, cuja definição é detalhada nas seções seguintes. Este parâmetro indica a capacidade do agente em tomar decisões melhores em situações similares no futuro, o que reflete o seu desempenho geral.
- Experiência E: Formada pelo conjunto estado atual s_t, ação aplicada ao sistema a_t, estado resultante da ação s_{t+1} e recompensa resultante r_t.

Assim, tem-se um problema de aprendizado de máquina que será abordado utilizando aprendizado por reforço como uma alternativa aos métodos tradicionais de controle. Essa alternativa pode ser justificada pela presença de não linearidades no modelo, o que dificulta o emprego de métodos tradicionais de controle. Para melhor compreensão do escopo deste tipo de aprendizado serão definidos os três principais tipos de aprendizado de máquina nas seções seguintes.

3.1 Aprendizado Supervisionado

Neste tipo de aprendizado, os dados fornecidos para o treinamento do algoritmo são previamente rotulados. Ou seja, o modelo irá aprender através de exemplos e depois deve ser capaz de generalizar o aprendizado em dados desconhecidos (GÉRON, 2022).

O aprendizado supervisionado pode ser aplicado em problemas de classificação, cujo objetivo é inferir a qual classe estes dados pertencem.

Outra aplicação importante é a regressão, na qual busca aproximar uma relação que represente o comportamento entrada-saída de um processo.

3.2 Aprendizado Não Supervisionado

Nesta forma de aprendizado o objetivo do algoritmo é descobrir padrões nos dados fornecidos sem a ajuda de um supervisor, ou seja, os dados não são previamente rotulados e o modelo deverá aprender possíveis padrões através da análise dos atributos.

Uma das maneiras de se conseguir encontrar padrões nos dados é através da *clusterização*. Esse método consiste na separação dos dados em conjuntos menores denominados *clusters*, que compartilham algumas semelhanças que são definidas através da configuração de parâmetros no algoritmo (GÉRON, 2022).

3.3 Aprendizado por Reforço

No aprendizado por reforço deve-se aprender qual ação a ser tomada em uma situação visando maximizar o acúmulo de uma recompensa numérica (SUTTON; BARTO, 2018). Para isso, é utilizado um sistema de exploração por tentativa e erro, e um sistema de recompensas baseado no objetivo do agente aprendiz. Apesar de não exigir supervisão, este tipo de aprendizado difere do aprendizado não supervisionado por realizar o aprendizado a partir de interações com o ambiente.

3.3.1 Estrutura do aprendizado baseado em reforço

Em um problema de aprendizado por reforço, têm-se os seguintes elementos (SUTTON; BARTO, 2018):

- Agente: É a entidade que irá receber informações do ambiente e tomar decisões, aplicando ações que irão resultar em mudanças no estado.
- Ambiente: Engloba tudo que é externo ao agente, ou seja, tudo que o agente não pode alterar arbitrariamente, como, por exemplo, o estado em que o sistema se encontra em determinado momento.

- Estado s: Conjunto de variáveis que descrevem o sistema em um determinado momento. A escolha do estado para um problema depende do escopo do projeto. No aeropêndulo, o estado será definido pelo ângulo e pela velocidade angular da haste.

- Ação a: É o processo iniciado pelo agente para causar uma mudança de estado no ambiente. A ação pode, por exemplo, ser um incremento de aceleração em um motor ou a abertura de uma válvula de escoamento em um tanque. No aeropêndulo a ação será incrementar ou decrementar a velocidade de rotação do motor sem escovas.

- Política π: Indica o conjunto das probabilidades do agente tomar determinada ação dado um estado. Ou seja, a política serve de base para determinar qual ação será tomada em um estado.

- Recompensa r: É o parâmetro utilizado para definir o objetivo do agente e indica se as ações tomadas podem ser consideradas "boas" ou "ruins". Tipicamente, se define o objetivo do agente é maximizar a recompensa total acumulada através das ações implementadas.

Assim, dado um estado s_t no instante t, o agente irá escolher a ação a_t de acordo com a política π. Tal ação é aplicada ao ambiente gerando um novo estado s_{t+1} e a recompensa associada r_{t+1}. Então, esse ciclo é reiniciado. Esta dinâmica está ilustrada na Figura 10.

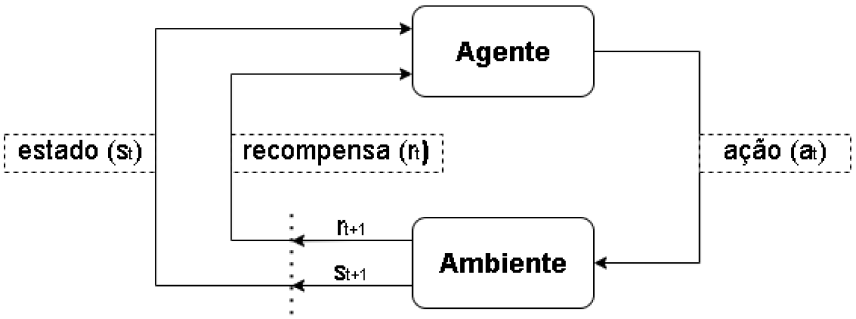

Figura 10 – Diagrama de interação entre agente e ambiente em um problema de aprendizado por reforço (SUTTON; BARTO, 2018).

Como já dito, o agente visa maximizar o acúmulo de recompensas ao longo do tempo, o que é denominado retorno. Na forma mais simples o retorno pode ser definido como o somatório das recompensas obtidas ao longo de um episódio. Isto é,

$$R_t = r_{t+1} + r_{t+2} + r_{t+3} + ... + r_f \tag{11}$$

em que f indica o tempo final do processo. Existem dois tipos de processos que são considerados no aprendizado por reforço:

1. Processos episódicos: Neste tipo de dinâmica, o número de incrementos temporais é finito. Quando o tempo final é atingido (ou uma condição terminal previamente definida), considera-se que o episódio foi concluído.

2. Processos contínuos: Não há tempo final definido. Logo, o sistema opera continuamente até que uma condição terminal seja atingida.

No caso dos processos contínuos, é possível que o retorno não convirja. Para eliminar esse problema, pode-se incluir um fator de desconto γ nas recompensas futuras (SUTTON; BARTO, 2018):

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \tag{12}$$

em que $0 \leq \gamma \leq 1$. O desconto irá alterar o impacto das recompensas futuras no cálculo do retorno. Adotando $\gamma < 1$, é possível tratar processos contínuos como se fossem finitos já que o somatório das recompensas converge.

3.3.2 Função Valor

A função valor pode ser empregada para mensurar a eficiência de uma política a partir de um estado e, desta forma, guiar o agente na escolha de uma ação (SUTTON; BARTO, 2018). Matematicamente, o retorno esperado ao se seguir uma política π a partir de um estado s_t, isto é, o valor de uma política π no estado s_t é dado por

$$V^{\pi}(s_t) = E_{\pi}\{R_t | s_t = s\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\} \tag{13}$$

É possível reescrever (13) obtendo uma equação de recorrência. Isso pode ser feito como mostrado a seguir

$$\begin{aligned} V^{\pi}(s_t) &= E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\right\} \\ &= E_{\pi}\left\{r_{t+1} + \gamma . \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} | s_t = s\right\} \\ &= E_{\pi}\left\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) | s_t = s\right\} \end{aligned} \tag{14}$$

A equação (14) é dita equação de Bellman para a função valor. Essa equação relaciona o valor de um determinado estado com a recompensa obtida ao seguir uma determinada política π, bem como com o valor do estado subsequente.

De forma similar, define-se o valor Q como a estimativa do retorno esperado ao se tomar uma ação arbitrária a_t em determinado estado s_t e depois seguir a política π, isto é, a cada ação é atribuído um valor Q e este valor é utilizado para otimizar a política adotada (WATKINS, 1989). Este processo é representado pela função valor de ação $Q^\pi(s_t, a_t)$, matematicamente dada por

$$Q^\pi(s_t, a_t) = E_\pi\{r_{t+1} + \gamma Q^\pi(s_{t+1}, a_{t+1})|s_t = s\} \tag{15}$$

3.3.3 Diferença Temporal

Diferença Temporal (DT) é um método para estimar a função de valor sem utilizar a expectativa de retorno e sem a necessidade de um modelo do ambiente (SUTTON; BARTO, 2018). Para isto, é utilizado um recurso chamado *bootstrapping* que consiste na utilização de um número n de estimativas calculadas anteriormente para realizar o cálculo da estimativa atual. No caso mais simples,

$$V^\pi(s_t) = V^\pi(s_t) + \alpha[r_{t+1} + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)] \tag{16}$$

em que o termo $[r_{t+1} + \gamma.V(s_{t+1}) - V(s_t)]$ é chamado de *TD-error* e indica a diferença entre o valor do estado atual e uma estimativa desse valor com base na recompensa atual e no valor do estado seguinte. Já α representa a taxa de aprendizagem, ou seja, é o parâmetro que vai definir o impacto de *TD-error* no cálculo de $V^\pi(s_t)$. Desta forma, aplicando uma ação a_t no ambiente e recebendo os valores de s_{t+1} e r_t, a estimativa do estado será atualizada *online* (SUTTON; BARTO, 2018).

3.3.4 *Q-Learning*

Este método consiste em calcular uma função valor de ação $Q^\pi(s_t,a_t)$ que se aproxime da função valor de ação ótima $Q^{\pi*}(s_t,a_t)$ (SUTTON; BARTO, 2018). Na prática, isto significa que o incremento no valor de $Q^\pi(s_t, a_t)$ irá considerar o maior valor do estado seguinte dentre todos os valores associados às ações possíveis, conforme indicado na expressão $\gamma \max_{a\in\mathcal{A}}[Q^\pi(s_{t+1}, a_t)]$. Por adotar a ação que maximiza o *Q-value* no próximo estado ao invés de seguir a política original π, este método é dito *off-policy*, ou seja, é um método de aprendizado que independe da política.

Na forma mais simples do algoritmo *Q-learning*, o *one-step Q-Learning*, a atualização é realizada com base somente no estado subsequente, ou seja, para atualizarmos o valor de $Q^{\pi}(s_{t-1}, a_{t-1})$ é necessário que o algoritmo tenha acesso apenas aos valores estimados de $Q^{\pi}(s_t, a_t)$ e de r_t.

Assim, a equação (17), que representa este algoritmo, segue uma estrutura análoga à função de valor definida previamente. Em suma, é calculado o *TD-error* entre o maior valor dentre as estimativas de Q quando $s = s_t$, e o valor de Q quando $s = s_{t-1}$. Em seguida o *TD-error* calculado é multiplicado pela taxa de aprendizagem α, e isto resultará no incremento utilizado para atualizar $Q^{\pi}(s_{t-1}, a_t)$. Matematicamente

$$Q^{\pi}(s_{t-1}, a_t) = Q^{\pi}(s_{t-1}, a_t) + \alpha[r_t + \gamma \max_{a \in \mathcal{A}} Q^{\pi}(s_t, a) - Q^{\pi}(s_{t-1}, a_t)] \tag{17}$$

Visto que $Q^{\pi}(s_t, a_t)$ busca maximizar seu valor independente da política adotada, é necessário que o agente adote uma técnica de exploração eficiente. Para isto, este projeto utiliza uma política chamada $\epsilon - greedy$. Nessa política, ou o agente escolhe a ação com o maior valor de Q com probabilidade $1-\epsilon$, ou escolhe uma ação aleatória com probabilidade ϵ.

Assim, escolhendo-se valores de ϵ adequados, esta política permite que o agente explore novas ações enquanto ainda prioriza a escolha da melhor ação em um estado s_t. O diagrama ilustrado na Figura 11 representa o funcionamento da política $\epsilon - greedy$.

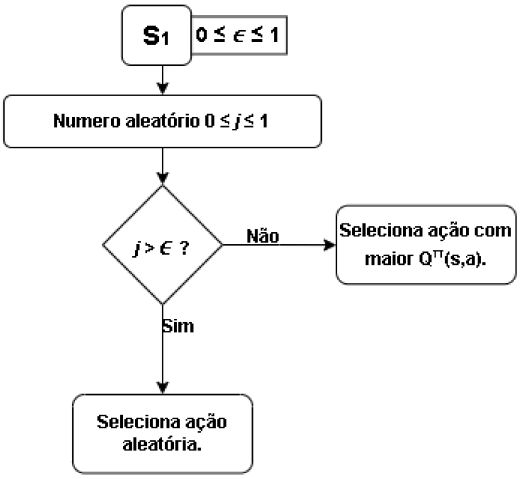

Figura 11 – Ilustração do funcionamento da política ϵ-*greedy*.

O algoritmo de aprendizado por reforço utilizando $\epsilon - greedy$ e *Q-Learning* está ilustrado na Figura 12.

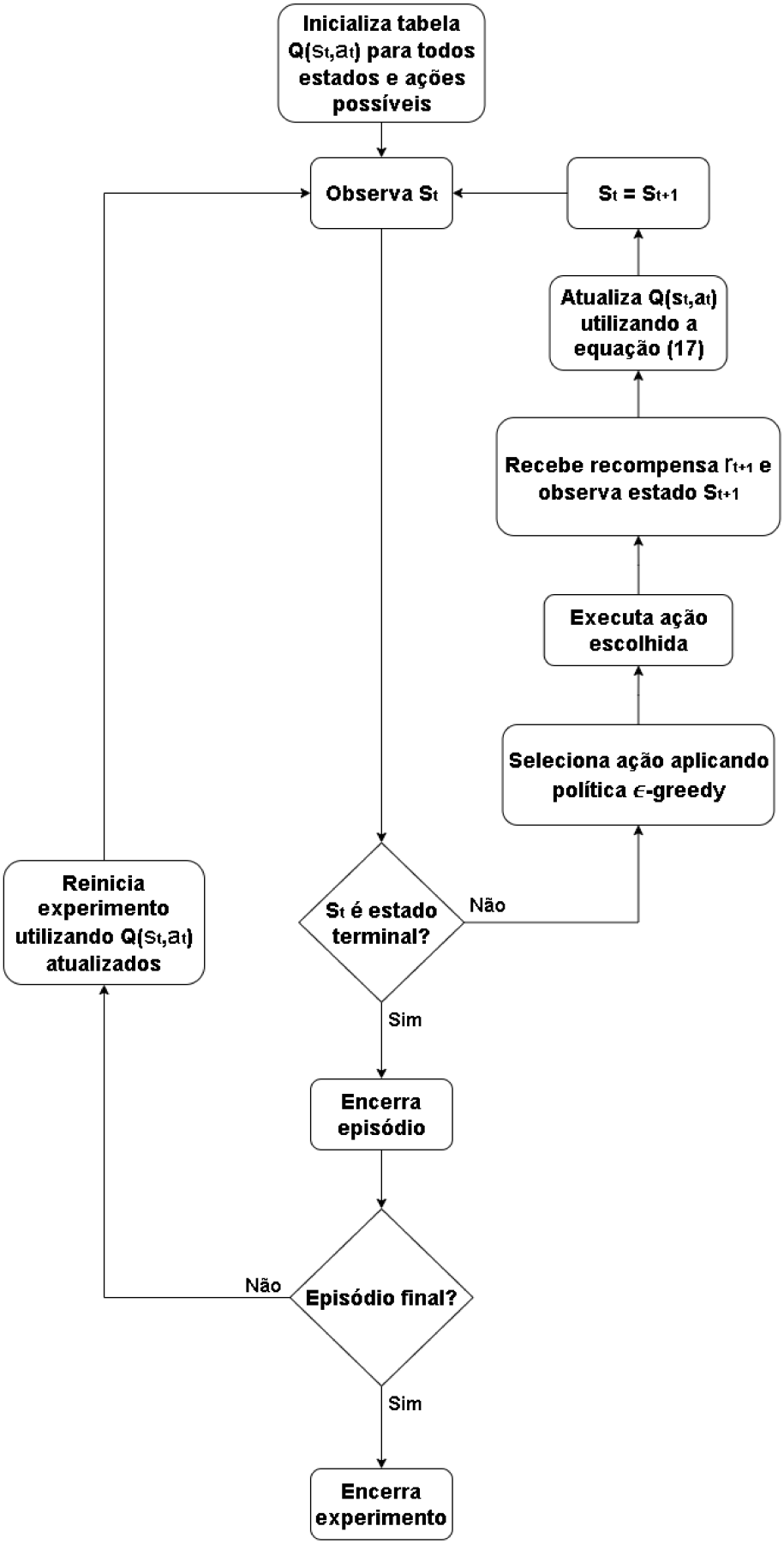

Figura 12 – Diagrama *Q-learning* utilizando a política ϵ-*greedy*.

Capítulo 4

RESULTADOS

Neste trabalho, implementou-se o *Q-learning* para controlar o aeropêndulo descrito no Capítulo 2. Para isso, simulações foram realizadas no modelo não linear. A metodologia para desenvolvimento dessas simulações e os resultados alcançados são apresentados na sequência.

4.1 Material utilizado

4.1.1 Hardware

Foi utilizado um computador pessoal com as seguintes características:

- CPU AMD Athlon com *clock* de 3,50 GHz.
- Memória RAM de 8 GB.
- Armazenamento SSD SATA com capacidade de 256 GB.
- Sistema Operacional Windows 10 com arquitetura de 64 bits.

4.1.2 Software

As simulações foram desenvolvidas utilizando a linguagem de programação Python na versão 3.10.4 e as seguintes bibliotecas:

- Scipy 1.8.0: Utilizada na resolução do sistema de equações diferenciais do modelo matemático do aeropêndulo.
- Numpy 1.22.3: Empregada para criar vetores e realizar manipulações algébricas.
- Matplotlib 3.5.2: Adotada na criação dos gráficos.

4.2 Arquitetura da simulação

A técnica de aprendizado por reforço descrita no Capítulo 2 somente pode ser aplicada caso os espaços de estados e ações sejam finitos. Sendo assim, definiu-se o ângulo mínimo como 0° e o máximo como 30°, e dividiu-se esse intervalo a cada 0,1°.

Para limitar o espaço de ações, considerou-se que o agente pode escolher uma das seguintes ações: Acelerar ou desacelerar. As magnitudes das acelerações/desacelerações variam de acordo com o módulo do erro de rastreamento $|e(t)| = |\theta_{ref}(t) - \theta(t)|$ (vide a Tabela 3), endo θ_{ref} o ângulo de referência. Essas magnitudes foram definidas heuristicamente.

Desta forma, se, por exemplo, $\theta = 3°$, $\omega^2 = 1000\frac{\text{rad}^2}{\text{s}^2}$ e a referência é $\theta_{ref} = 15°$, então $|e| > 5°$ e o agente irá aumentar ou diminuir (a depender de sua política) ω^2 em $800\frac{\text{rad}^2}{\text{s}^2}$.

Vale ressaltar que, nas simulações, as unidades de ângulo e velocidade de rotação estão no Sistema Internacional. Contudo, para apresentação dos resultados, optou-se por representar os ângulos em graus. Ademais, definiu-se um tempo de amostragem $T_a = 0{,}1$ s.

Tabela 3 – Valores dos incrementos aplicados a ω^2 de acordo com o módulo do erro de rastreamento.

Módulo do erro de rastreamento	Magnitude da aceleração/desaceleração ($\pm\frac{\text{rad}^2}{\text{s}^2}$)
$\lvert e\rvert \geq 5°$	800
$4° \leq \lvert e\rvert < 5°$	700
$3° \leq \lvert e\rvert < 4°$	650
$2{,}5° \leq \lvert e\rvert < 3°$	600
$2° \leq \lvert e\rvert < 2{,}5°$	500
$1{,}7° \leq \lvert e\rvert < 2°$	400
$1{,}5° \leq \lvert e\rvert < 1{,}7°$	300
$1{,}3° \leq \lvert e\rvert < 1{,}5°$	200
$1° \leq \lvert e\rvert < 1{,}3°$	180
$0{,}7° \leq \lvert e\rvert < 1°$	160
$0{,}5° \leq \lvert e\rvert < 0{,}7°$	140
$0{,}2° \leq \lvert e\rvert < 0{,}5°$	120
$0° \leq \lvert e\rvert < 0{,}2°$	1,5

Para que o agente alcance a tarefa de controle definida, adotou-se a seguinte função de recompensas:

$$r_t(\theta(t), \theta_{ref}) = \begin{cases} 1, & \text{se} \quad (\theta_{ref} - 0{,}5) \leq \theta(t) \leq (\theta_{ref} + 0{,}5), \\ -1, & \text{se} \quad \theta(t) = \theta_t(t), \\ 0 & \text{caso contrário.} \end{cases} \tag{18}$$

em que θ_t representa um estado terminal definido pelos limites do espaço de estados, ou seja, quando $\theta_t(t) = 0°$ ou $\theta_t(t) = 30°$. Nesse último caso, o episódio é reiniciado.

Com esta função, tem-se uma recompensa positiva, que irá aumentar o retorno do episódio, caso a referência seja alcançada, e uma recompensa negativa, que diminui o retorno, penalizando o agente quando um estado terminal é alcançado.

Para implementação das simulações, dividiu-se o código nas seguintes funções:

- ❏ din_aeropend: Estabelece os parâmetros do aeropêndulo e do algoritmo de aprendizado por reforço;
- ❏ def_recompensas: Define as recompensas de acordo com a referência;
- ❏ estado_terminal: Verifica se algum dos estados terminais foi alcançado;
- ❏ e_greedy: Aplica a política $\epsilon - greedy$ para escolha da ação;
- ❏ acao: Função que realiza o incremento na velocidade angular de acordo com a ação escolhida. Isto é feito através da criação de *buckets* de incremento que variam de acordo com o módulo do erro de rastreamento do ângulo: $e(t) = |\theta(t) - \theta_{ref}|$;
- ❏ controle: Implementa o algoritmo *Q-Learning* e atualiza a tabela de *Q-values*;
- ❏ planta: Função que simula o comportamento do aeropêndulo, definido por um período de amostragem de acordo com o estado atual s_t e com a ação atual a_t;
- ❏ plot_q: Produz um gráfico da tabela de *Q-values*;
- ❏ treino: Função de tratamento dos dados obtidos pelo treinamento.

O código foi versionado em um repositório da plataforma *Github*[1] e as instruções de utilização estão na página principal do mesmo. Além disso, a Figura 13 apresenta um diagrama simplificado da arquitetura que expressa o relacionamento entre as funções citadas, visando tornar o funcionamento do programa mais claro.

Nota-se que o código permite ao operador escolher entre dois métodos de atuação: Um método sem treinamento, com o algoritmo sendo aplicado uma única vez de forma que seja possível observar o impacto imediato na mudança dos hiperparâmetros.

Já no outro método é realizado um treinamento com um número de episódios escolhido pelo operador, assim serão realizados vários ciclos de melhoria nos hiperparâmetros obtendo-se uma tabela de *Q-values* otimizada que poderá ser utilizada posteriormente.

[1] https://github.com/ckmoraiz/pfc-q-learning

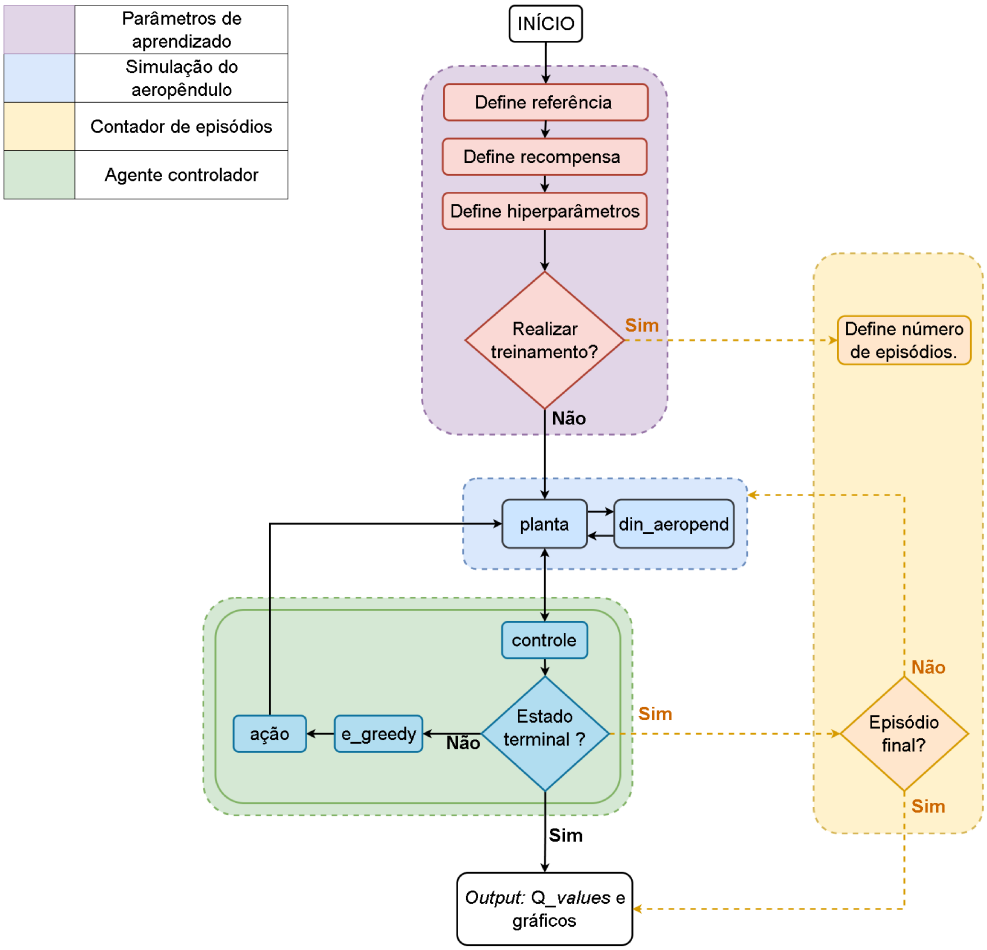

Figura 13 – Diagrama simplificado da arquitetura da solução proposta.

A Figura 14 mostra um exemplo do código em execução para um ângulo de referência de 15° e treinamento com 3 episódios. As condições iniciais foram configuradas considerando um protótipo partindo do repouso, ou seja, $\theta_0 = 0°$, $\omega_0 = 0\frac{\text{rad}}{\text{s}}$ e $\omega_0^2 = 0\frac{\text{rad}^2}{\text{s}^2}$.

```
Insira ângulo inteiro de referência (em graus):
Referência:15
Tabela de Q_values encontrada e carregada com sucesso!

Escolha:
[1]Simulação única.
[2]Simulação de treino com N episódios.
:2
Insira quantidade N de episódios para iterações:
Episódios:3
Episódio 1 concluído!
Episódio 2 concluído!
Episódio 3 concluído!
epsilon:0.9, gamma:0.4, alpha:0.9
```

Figura 14 – Exemplo de interação com o programa via terminal.

Assim, com o programa de simulação desenvolvido, é possível testar o impacto de variações nos parâmetros de ajuste no aprendizado do agente. Isso será realizado a seguir.

4.3 Análise de resultados

Nesta seção, discutir-se-á a escolha dos parâmetros de treinamento e verificar-se-á seu impacto no desempenho quanto ao rastreamento de referências constantes para o ângulo da haste. Esse estudo está dividido em três partes: Primeiro é realizada uma análise do número de episódios de treinamento necessários até que o agente aprenda a guiar a haste para a referência θ_{ref}. Em seguida, investigam-se os efeitos de variações nos parâmetros de ajuste no aprendizado. Por fim, são realizadas simulações variando-se a referência para validar a capacidade de generalização do agente.

4.3.1 Análise do número de episódios de treinamento

Nesta etapa foram realizadas simulações com $\theta_{ref} = 15°$ e condições iniciais nulas. Mais ainda, o número de episódios foi variado entre 500 e 5000 com incrementos de 500. A Figura 15 mostra a média dos ângulos da haste durante o último episódio, isto é, já após o treinamento. Por simplicidade, adotaram-se todos os hiperparâmetros (ϵ, γ, α) como 0,5.

Vale comentar que os valores apresentados nas Figuras 15 e 16 são as médias dos ângulos de saída ao longo do último episódio (incluindo o regime transitório). Portanto, se, por exemplo, o agente estava sendo treinado com 1000 episódios, a média de todos os ângulos obtidos no episódio 1000 foi utilizada para gerar os gráficos analisados.

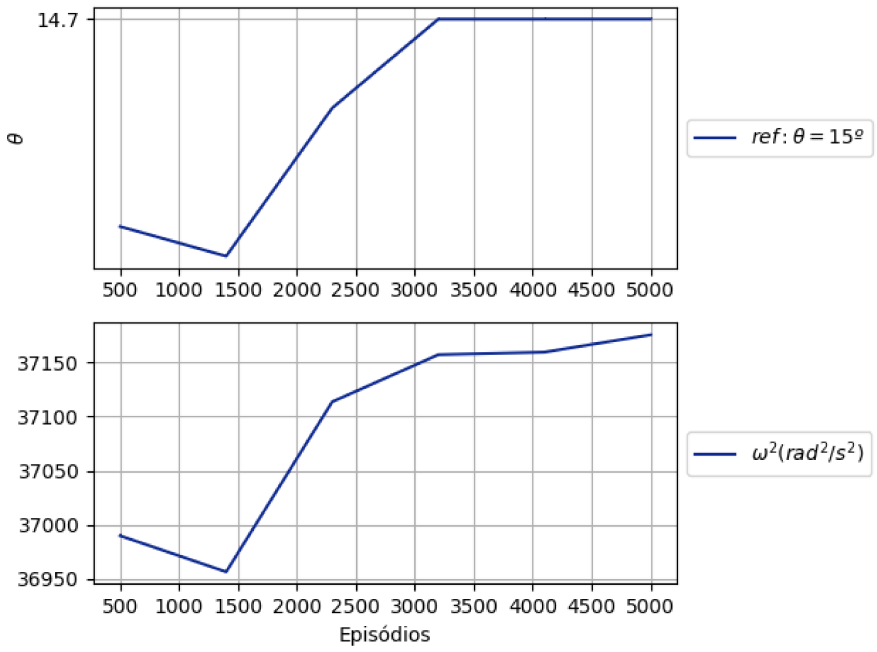

Figura 15 – Análise do impacto do número de episódios no rastreamento da referência.

Nota-se que, após aproximadamente 3000 episódios, o agente guiou a haste para um ângulo médio de 14,7°, indicando que utilizar uma quantidade maior de episódios não

resultará em diminuição do erro em regime estacionário. A Figura 16 mostra o resultado de uma etapa de refinamento para identificar qual a menor quantidade de episódios necessária para a convergência da haste para próximo da referência.

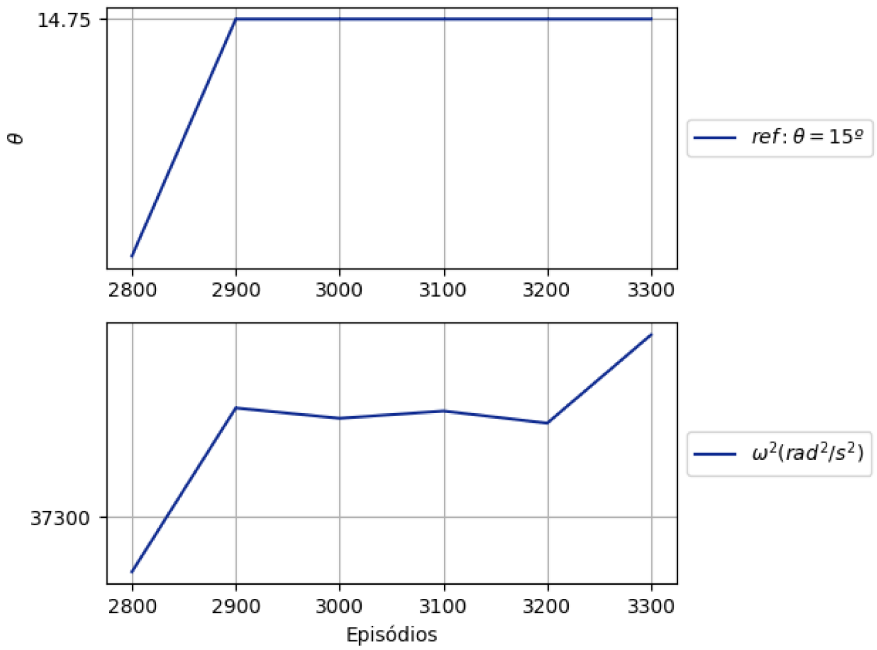

Figura 16 – Refinamento da análise do número de episódios para treinamento do agente.

É possível verificar que após 2900 episódios não ocorrem mudanças significativas no comportamento do sistema e a média do ângulo da haste no último episódio converge para 14,75°, resultando em um erro de rastreamento de aproximadamente 1,6% (0,25°). Portanto, nas análises posteriores, os treinamentos serão executados com 2900 episódios.

4.3.2 Análise da sensibilidade

A análise de sensibilidade refere-se à investigação dos impactos de variações nos hiperparâmetros do *Q-learning* no aprendizado do agente em realizar a tarefa. Especificamente, essa análise será realizada em três etapas, como descrito a seguir.

- ❏ Primeira etapa:
  - . $\epsilon = [0{,}25;\ 0{,}5;\ 0{,}75]$;
  - . $\gamma = 0{,}5$;
  - . $\alpha = 0{,}5$.
- ❏ Segunda etapa:
  - . $\epsilon =$ Melhor resultado da etapa anterior;
  - . $\gamma = [0{,}25; 0{,}5; 0{,}75]$;
  - . $\alpha = 0{,}5$.

❏ Terceira etapa:

. ϵ = Melhor resultado da primeira etapa;
. γ = Melhor resultado da etapa anterior;
. $\alpha = [0{,}25; 0{,}5; 0{,}75]$.

As Figuras 17, 18 e 19 apresentam as curvas obtidas para θ e ω^2 no último episódio do treinamento variando-se os parâmetros, como detalhado anteriormente. A referência e as condições iniciais adotadas foram as mesmas da seção anterior. O número de episódios foi definido como 2900. Para auxiliar na comparação dos resultados adotar-se-á a somatória do módulo do erro E dada por

$$E = \sum_{t=0}^{300} |e(t)| \tag{19}$$

Para cada etapa, os seguintes valores de E foram obtidos:

❏ Primeira etapa:

1. $E_{\epsilon 1} = 874{,}2$
2. $E_{\epsilon 2} = 642{,}0$
3. $E_{\epsilon 3} = 580{,}2$

❏ Segunda etapa:

1. $E_{\gamma 1} = 548{,}1$
2. $E_{\gamma 2} = 454{,}8$
3. $E_{\gamma 3} = 499{,}3$

❏ Terceira etapa:

1. $E_{\alpha 1} = 486{,}9$
2. $E_{\alpha 2} = 445{,}6$
3. $E_{\alpha 3} = 435{,}6$

Portanto, analisando quantitativamente, os parâmetros que apresentaram um melhor desempenho são: $\epsilon = 0{,}75$, $\gamma = 0{,}5$ e $\alpha = 0{,}75$.

Analisando qualitativamente, na primeira etapa, o ângulo da haste é levado para o entorno da referência mais rapidamente com $\epsilon = 0{,}75$. Isso era esperado, já que esta variável regula a frequência com que o agente explora novas ações em um certo estado, ou seja, valores mais elevados indicam que ações que maximizam o valor Q serão escolhidas, favorecendo a experiência já adquirida e, consequentemente, apresentando respostas mais rápidas. Portanto, este valor foi utilizado na etapa seguinte. Já na segunda etapa, os resultados ficaram similares. Contudo, adotando-se $\gamma = 0{,}5$ obtém-se uma resposta menos

oscilatória. Por fim, na última etapa, $\alpha = 0{,}75$ apresentou o melhor resultado, indicando que valores mais elevados da taxa de aprendizagem podem tornam o aprendizado mais rápido para a tarefa de controle aqui considerada. Ademais, constata-se que, quanto maior o valor de α, menor é o erro em regime estacionário (i.e. mais a média de θ converge para próximo da referência de 15°).

Após esta análise, foi realizada uma etapa de refinamento adicional dos hiperparâmetros resultando em $\epsilon = 0{,}9$, $\gamma = 0{,}4$, $\alpha = 0{,}9$. O índice de desempenho E para esse conjunto de hiperparâmetros foi 413,3, o que representa uma melhoria de 52,7% em relação ao maior erro observado na primeira etapa ($E = 874{,}2$).

O desempenho do agente treinado com esse conjunto de parâmetros é apresentado na Figura 20.

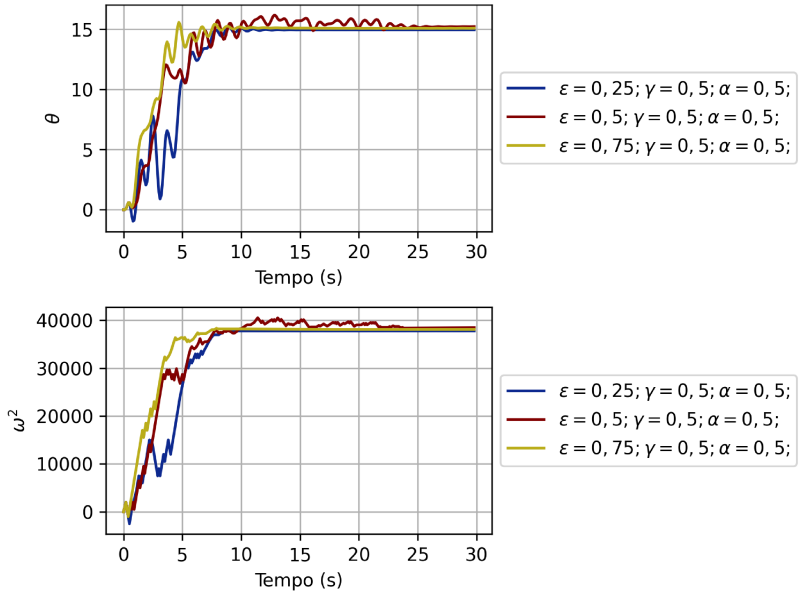

Figura 17 – Desempenho do agente no último episodio de treinamento para diferentes valores de ε.

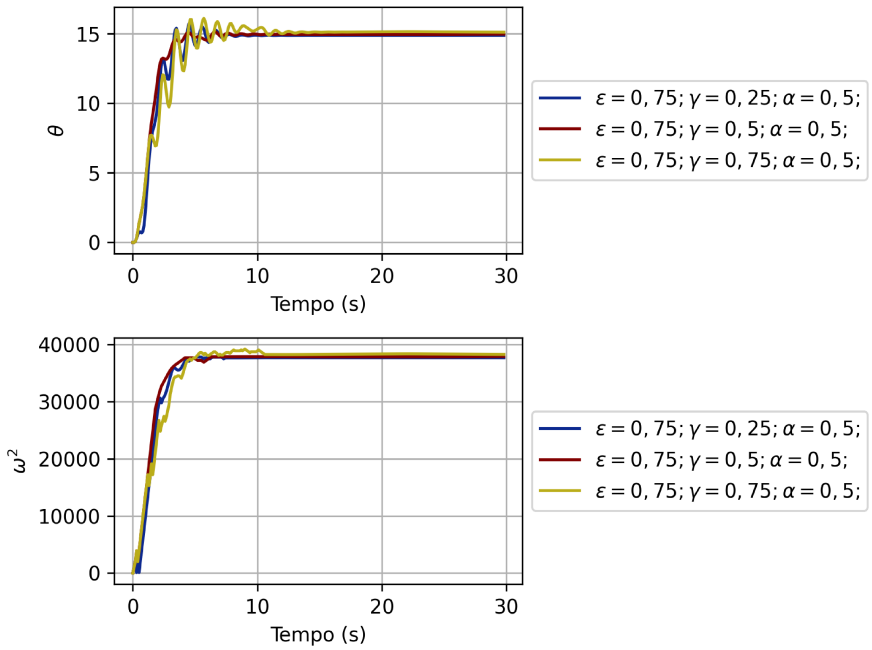

Figura 18 – Desempenho do agente no último episodio de treinamento para diferentes valores de γ.

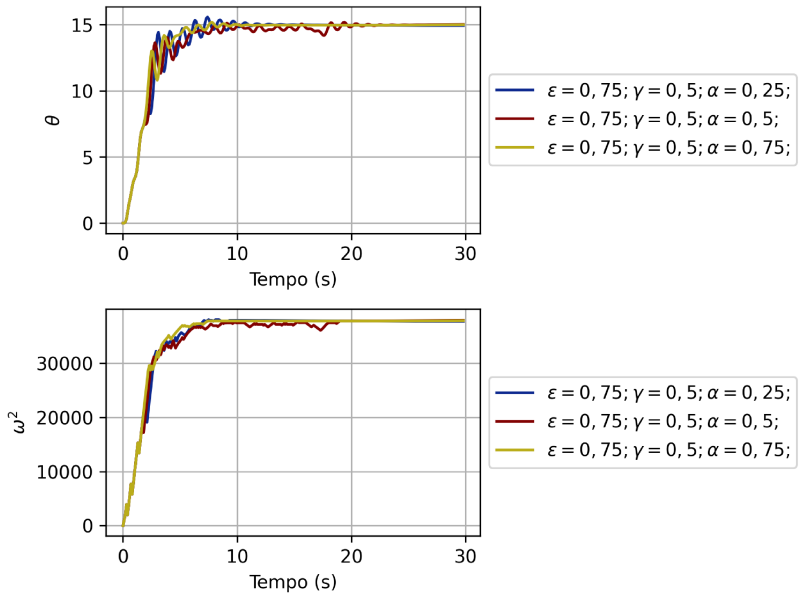

Figura 19 – Desempenho do agente no último episodio de treinamento para diferentes valores de α.

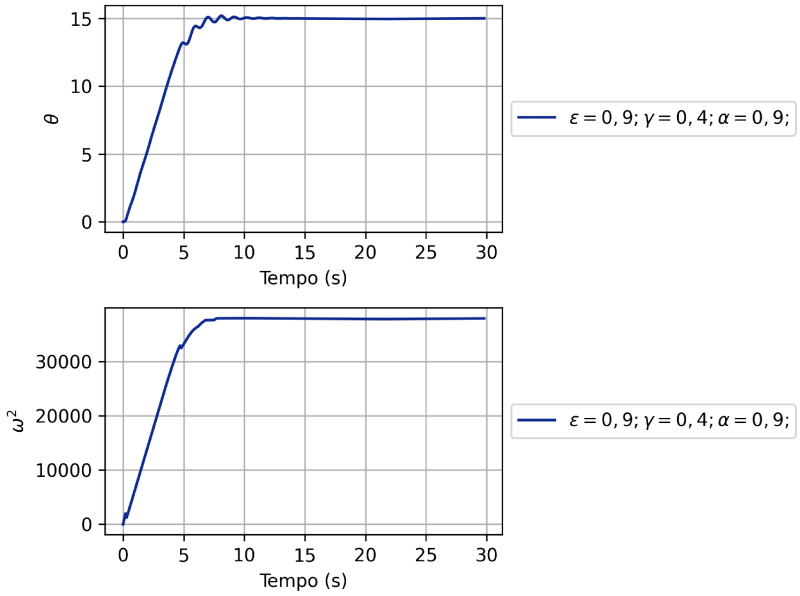

Figura 20 – Desempenho do agente utilizando os parâmetros definidos após o refinamento.

Observando os resultados anteriores, é possível perceber que, ao final do treinamento, o agente sempre foi capaz de levar a haste do aeropêndulo para a referência de 15°. A seguir, é realizada uma análise variando-se o ângulo de referência.

4.3.3 Análise de resultados para diferentes referências

Por tentativa e erro, as variações da velocidade de rotação de acordo com o módulo do erro de rastreamento mostradas na Tabela 3 foram refinadas. Os valores finais são exibidos na Tabela 4.

Tabela 4 – Valores dos incrementos aplicados a ω^2 de acordo com o erro de rastreamento.

Faixa do erro de rastreamento	Magnitude da aceleração/desaceleração ($\pm\frac{\text{rad}^2}{\text{s}^2}$)
$\lvert e\rvert \geq 5°$	1400
$4° \leq \lvert e\rvert < 5°$	900
$3° \leq \lvert e\rvert < 4°$	850
$2{,}5° \leq \lvert e\rvert < 3°$	800
$2° \leq \lvert e\rvert < 2{,}5°$	700
$1{,}7° \leq \lvert e\rvert < 2°$	600
$1{,}5° \leq \lvert e\rvert < 1{,}7°$	460
$1{,}3° \leq \lvert e\rvert < 1{,}5°$	380
$1° \leq \lvert e\rvert < 1{,}3°$	360
$0{,}7° \leq \lvert e\rvert < 1°$	340
$0{,}5° \leq \lvert e\rvert < 0{,}7°$	320
$0{,}2° \leq \lvert e\rvert < 0{,}5°$	160
$0° \leq \lvert e\rvert < 0{,}2°$	1,5

Para mostrar que esse ajuste melhorou o desempenho do sistema, apresenta-se, na Figura 21, o resultado comparativo entre a utilização dos valores das Tabelas 3 e 4. Observando a figura, nota-se uma melhoria no cumprimento da tarefa de controle. Para validar tal melhoria, adotou-se como parâmetro quantitativo o erro acumulado E, definido anteriormente.

Assim, obteve-se $E_{T3} = 413{,}3$ e $E_{T4} = 269{,}6$, para os gráficos que utilizaram os valores da Tabela 3 e 4, respectivamente. Ou seja, o refinamento heurístico dos valores utilizados para variar a velocidade de rotação do motor resultou em uma redução de 34,7% do índice de desempenho E.

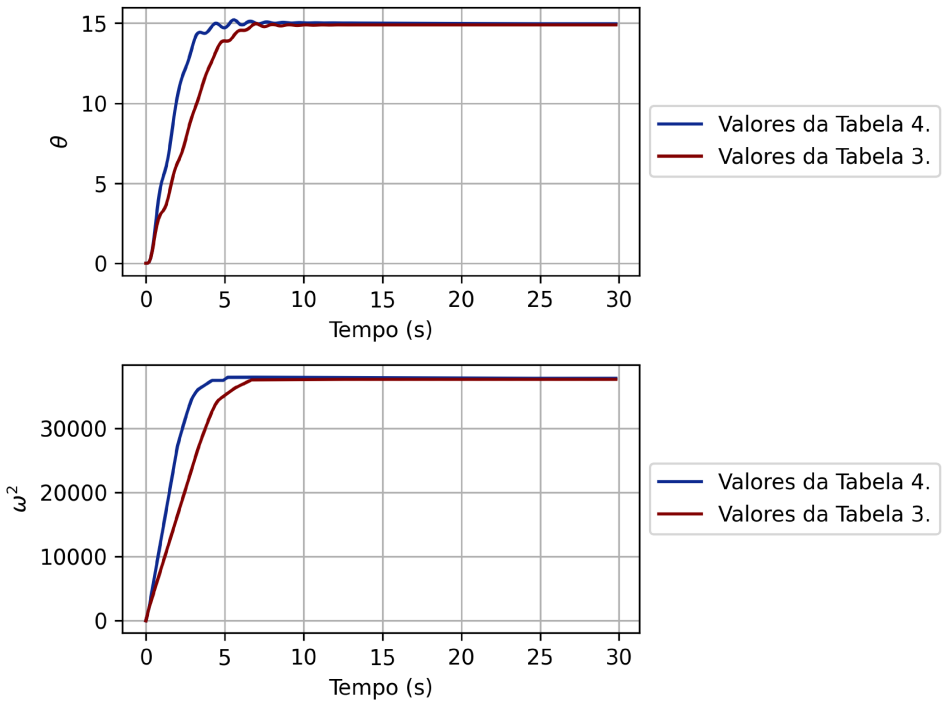

Figura 21 – Comparação da resposta do sistema após o treinamento utilizando os incrementos de velocidade de rotação iniciais (Tabela 3) e após o refinamento (Tabela 4).

Enquanto nas simulações anteriores foi considerada apenas a referência de 15°, testes de rastreamento foram realizados para outros ângulos de referência. Especificamente, considerou-se $\theta_{ref} \in \{5°,\ 10°,\ 15°,\ 20°,\ 25°\}$. Verificou-se que o sistema proposto é capaz de aprender a guiar o agente para cada uma dessas referências. As Figuras de 22 a 26 mostram o desempenho do sistema após o treinamento para os hiperperâmetros e condições iniciais definidos anteriormente. Pode-se constatar que o agente aprendeu a guiar a haste para todas essas referências. Isso ilustra uma vantagem do aprendizado por reforço no controle de sistemas não lineares.

Vale comentar que, caso fosse empregada alguma técnica de controle linear, um projeto de controlador para cada posição de equilíbrio deveria ser realizado.

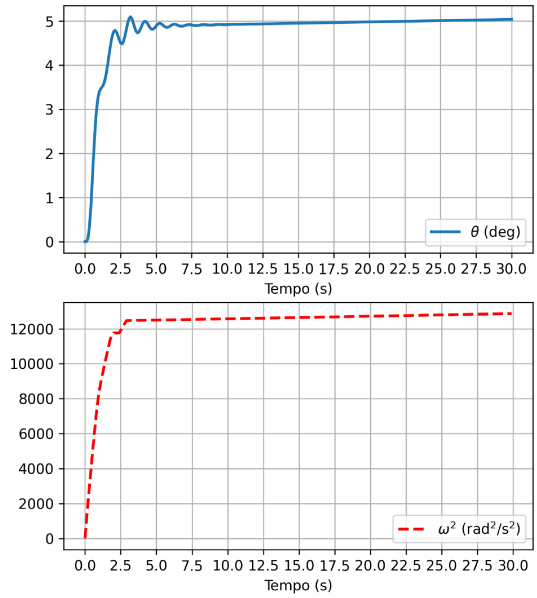

Figura 22 – Simulação após o treinamento para a referência de 5°.

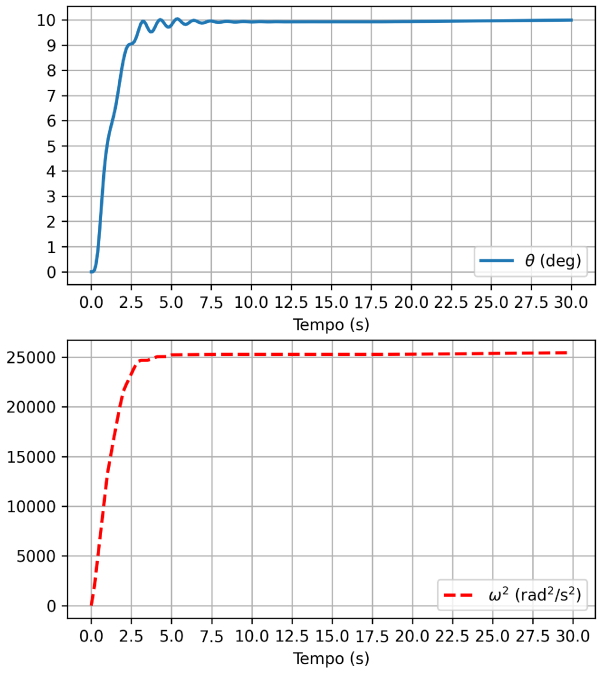

Figura 23 – Simulação após o treinamento para a referência de 10°.

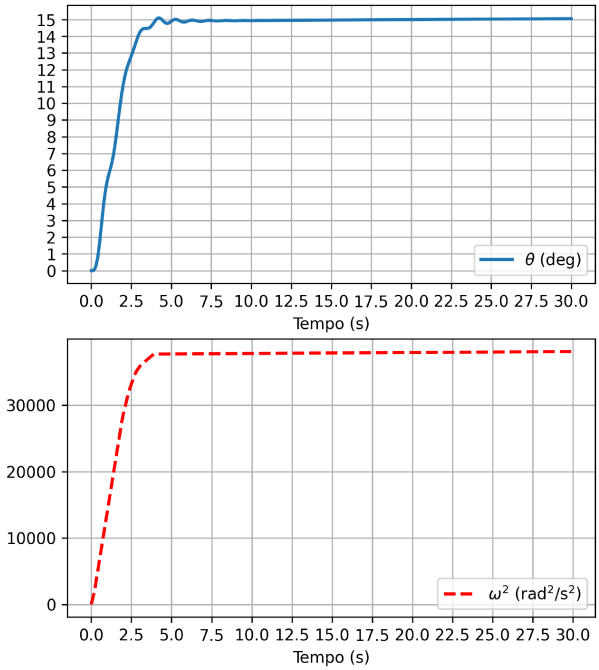

Figura 24 – Simulação após o treinamento para a referência de 15°.

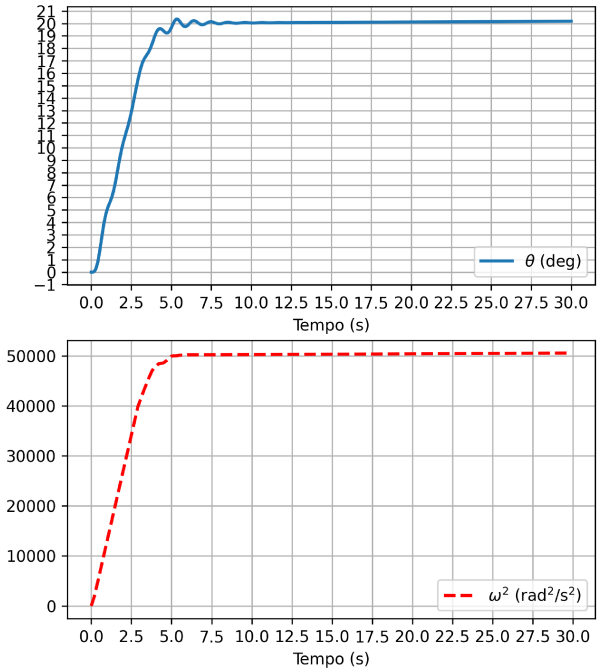

Figura 25 – Simulação após o treinamento para a referência de 20°.

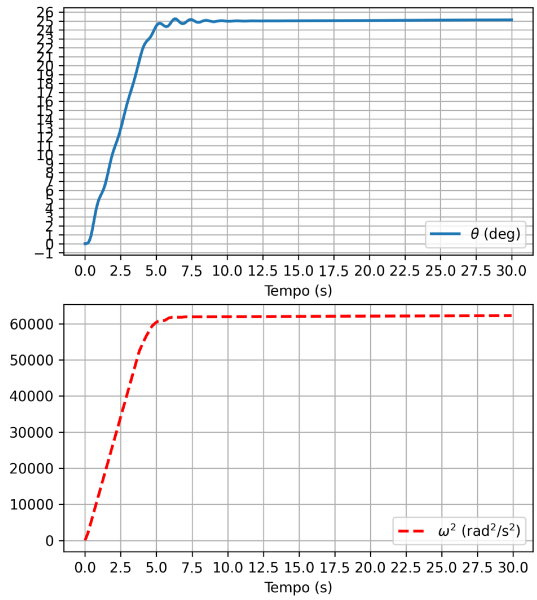

Figura 26 – Simulação após o treinamento para a referência de 25°.

Capítulo 5

Conclusão

O objetivo deste trabalho é aplicar um método de aprendizado de máquina no controle do aeropêndulo. Para isto, foi desenvolvida uma simulação computacional do modelo matemático não linear da planta e implementou-se o *Q-learning* para atingir a tarefa de controle (guiar a haste até uma posição de referência).

Para implementação do *Q-learning*, foi definido um espaço de estados e um espaço de ações. Especificamente considerou-se um intervalo entre 0° e 30°, espaçado a cada 0,1°, para o ângulo da haste. O espaço de ações foi limitado considerando-se duas ações: acelerar ou desacelerar. Ao decidir por uma destas ações, o agente varia o quadrado da velocidade de rotação do motor. A magnitude dessas variações é relacionada com o módulo do erro de rastreamento.

Além disso, foi definida uma função de recompensas $r_t(\theta(t), \theta_{ref})$ para guiar o aprendizado do agente. Essa função estabeleceu uma recompensa positiva quando a haste atinge a referência, e uma recompensa negativa caso o episódio seja terminado. Assim, quando os valores de Q foram atualizados com tais recompensas, o agente foi capaz de aprender, através da interação com o ambiente, quais ações produziam o melhor resultado.

Nas simulações, três tipos de análises foram realizadas:

- ❏ Análise do número de episódios de treinamento;
- ❏ Análise de sensibilidade do aprendizado em relação aos parâmetros de ajuste;
- ❏ Análise de resultados para diferentes referências.

Para a análise do número de episódios necessários ao treinamento, o agente foi treinado com diferentes números de episódios que variavam entre 500 e 5000, com incrementos de 500 episódios. Durante essa análise, consideraram-se $\theta_{ref} = 15°$, condições iniciais nulas, e calculou-se a média dos ângulos durante o último episódio de cada treinamento. Após um refinamento desta análise, notou-se que o ângulo da haste do aeropêndulo converge para a referência após aproximadamente 2900 episódios. Ou seja, notou-se que o agente aprende a realizar a tarefa de controle após esse número de episódios.

Então, foi realizada a análise de sensibilidade os parâmetros de ajuste do *Q-learning*. Nesse cenário, a taxa de desconto γ, a taxa de aprendizado α e o limite definido para a política ϵ, foram variados em intervalos pré-definidos. Como resultado, pode-se verificar os impactos das variações nesses parâmetros no aprendizado do agente. Constatou-se que, ao variar ϵ e α, os valores mais próximos de 1 resultaram em um melhor desempenho. Já γ apresentou melhores resultados para valores próximos de 0,5. Ao final da análise, foi possível encontrar um conjunto de parâmetros que apresentava um desempenho adequado, isto é, que resultava no agente aprender a levar a haste para a posição de referência com um comportamento transitório razoável.

Também foram realizadas simulações para diferentes valores de referência. Mais especificamente, considerou-se $\theta_{ref} \in \{5°,\ 10°,\ 15°,\ 20°,\ 25°\}$. Observou-se que o agente foi capaz de levar a haste para a referência em todos os casos ao final do treinamento, indicando uma vantagem de métodos de aprendizado por reforço no controle de sistemas não lineares. Mais precisamente a não necessidade de se reprojetar o controlador de acordo com a condição de operação.

Por fim, para possíveis trabalhos futuros pode ser considerado o estudo de técnicas mais avançadas de aprendizado de máquina que combinem aprendizagem por reforço e *deep learning* como, por exemplo, o *Deep Q-Learning*. Ademais, uma validação experimental do sistema de controle desenvolvido seria interessante.

Referências

BROWN, G. **Discovering the STM32 microcontroller**. Bloomington: Indiana University, 2016.

DORF, R. C.; BISHOP, R. H. **Sistemas de Controle Modernos**. 3ª ed., Rio de Janeiro: Editora LTC, 2001.

GARCIA, C. **Modelagem e Simulação de Processos Industriais e de Sistemas Eletromecânicos Vol. 1**. São Paulo: Edusp, 2005.

GÉRON, A. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow**. 3rd. ed. Sebastopol: O'Reilly Media, 2022.

MITCHELL, T. **Machine learning**. New York: McGraw-hill, 1997. v. 1.

MONARD, M. C.; BARANAUSKAS, J. A. **Conceitos Sobre Aprendizado de Máquina. Sistemas Inteligentes Fundamentos e Aplicações**. Barueri: Manole Ltda, 2003. 89-114 p.

NAGEL, L. **What is an Electronic Speed Controller & How Does an ESC Work**. 2022. Disponível em: <https://www.tytorobotics.com/blogs/articles/what-is-an-esc-how-does-an-esc-work>.

NEDELKOVSKI, D. **How Brushless DC Motor Works? BLDC and ESC Explained**. 2021. Disponível em: <https://howtomechatronics.com/how-it-works/how-brushless-motor-and-esc-work/>.

OGATA, K. **Engenharia de controle moderno**. São Paulo: Pearson Education, 2011.

STMICROELECTRONICS. **Medium-density performance line ARM-based interfaces datasheet**. 2022. Disponível em: <https://www.st.com/resource/en/datasheet/stm32f103c8.pdf>.

SUTTON, R. S.; BARTO, A. G. **Reinforcement learning: An introduction**. Cambridge: MIT press, 2018.

WATKINS, C. J. C. H. **Learning from delayed rewards**. Cambridge: King's College, 1989.