UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
ENGENHARIA AEROESPACIAL

RAFAEL CABRAL DOS SANTOS

INVESTIGAÇÃO DE UM SISTEMA DE CONTROLE PARA POUSO DE BOOSTERS UTILIZANDO APRENDIZADO DE MÁQUINA POR REFORÇO

Joinville
2024

RAFAEL CABRAL DOS SANTOS

INVESTIGAÇÃO DE UM SISTEMA DE CONTROLE PARA POUSO DE BOOSTERS UTILIZANDO APRENDIZADO DE MÁQUINA POR REFORÇO

Trabalho apresentado como requisito para obtenção do título de Bacharel em Engenharia Aeroespacial, no Centro Tecnológico de Joinville, da Universidade Federal de Santa Catarina.

Orientador: Juan Pablo de Lima Costa Salazar, Ph. D.

Coorientador: Dr. Rafael Gigena Cuenca

Joinville
2024

RAFAEL CABRAL DOS SANTOS

INVESTIGAÇÃO DE UM SISTEMA DE CONTROLE PARA POUSO DE BOOSTERS UTILIZANDO APRENDIZADO DE MÁQUINA POR REFORÇO

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do título de Bacharel em Engenharia Aeroespacial, no Centro Tecnológico de Joinville, da Universidade Federal de Santa Catarina.

Joinville (SC), 5 de julho de 2024.

Banca Examinadora:

Juan Pablo de Lima Costa Salazar, Ph. D.
Orientador/Presidente
UFSC

Prof. Dr. Rafael Gigena Cuenca
Coorientador
UFSC

Prof. Dr. Pablo Andretta Jaskowiak
Membro(a)
UFSC

Prof. Dr. Antônio Otaviano Dourado
Membro(a)
UFSC

RESUMO

O principal custo de uma missão orbital diz respeito ao primeiro estágio do veículo lançador. Atualmente, esse estágio é apenas parcialmente recuperado acarretando em altos custos operacionais do setor aeroespacial. Visando reduzir tais custos, procura-se recuperar o primeiro estágio pousando-o em solo verticalmente, através de sistemas de controle de estabilização autônomos. Controladores podem ser desenvolvidos a partir de metodologias clássicas ou modernas, que utilizam maior poder computacional para processar as informações. Este trabalho investiga uma abordagem recente para a obtenção de um controlador, visando mimetizar os resultados já alcançados por empresas como SpaceX e Blue Origin. Para isso, pretende-se modelar um sistema simplificado de controle para a etapa de pouso do booster utilizando aprendizado de máquina por reforço, a fim de definir diretrizes eficientes para os sistemas. Será construído um ambiente de simulação 2D em Python, utilizado a framework Box2D, para modelar a física do booster. Em seguida, serão comparados os modelos obtidos com dados de telemetria de um pouso da missão SpaceX CRS-11.

Palavra-chave: Aprendizado por reforço. Aprendizado de máquina. Booster. Teoria do controle.

ABSTRACT

The main cost of an orbital mission concerns the first stage of the launch vehicle. Currently, this stage is only partially recovered, resulting in high operational costs in the aerospace sector. To reduce these costs, efforts are being made to recover the first stage by landing it vertically on the ground using autonomous stabilization control systems. Controllers can be developed using either classical methodologies or modern approaches that leverage greater computational power to process information. This work investigates a recent approach to obtaining a controller, aiming to mimic the results already achieved by companies like SpaceX and Blue Origin. To this end, a simplified control system for the booster landing stage will be modeled using reinforcement learning to define efficient system guidelines. A 2D simulation environment will be built in Python using the Box2D framework to model the physics of the booster. Subsequently, the obtained models will be compared with telemetry data from a SpaceX CRS-11 mission landing.

Keywords: Reinforcement Learning. Machine Learning. Control theory. Booster.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Os benefícios das missões espaciais nas últimas décadas são inegáveis, impulsionando a humanidade a patamares de conectividade sem precedentes na história. Desde o cotiadiano de pessoas comuns, para geolocalização, até sua influência no sistema financeiro mundial (O'CONNOR et al., 2019), tecnologias presentes nos satélites alteraram o estilo da vida humana nas últimas décadas.

Esses são, em grande parte, resultados da corrida espacial durante a Guerra Fria, onde Estados Unidos e União Soviética disputaram a hegemonia de suas nações por meio da conquista do espaço (KHATAL et al., 2023). Contudo, apesar de todos os avanços do setor, um problema sempre se manteve presente, o custo operacional. Estima-se que uma única missão do programa espacial americano Apollo chegou a custar 3.69 bilhões de dólares, valor ajustado segundo a inflação desde 1973 (DREIER, 2022).

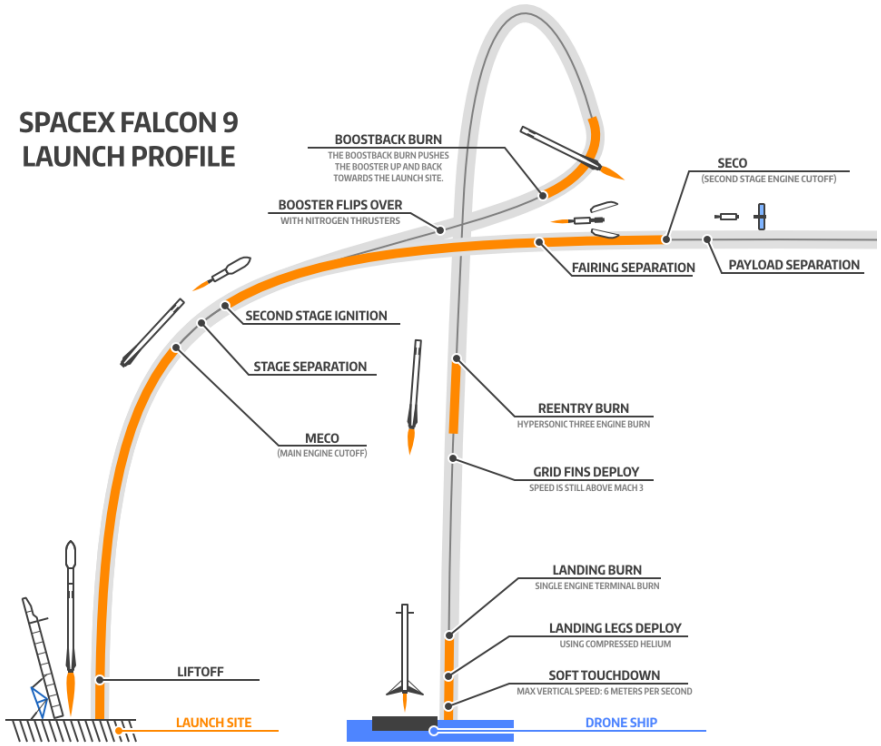
Nos dias presentes, segundo Tomanek e Hospodka (2018), os custos associados a uma missão orbital são predominantemente atribuídos ao veículo lançador, representando aproximadamente 90% do montante total, com cerca de 75% desse valor originando-se do primeiro estágio do foguete, conhecido como booster. Logo, visando viabilizar a presença humana em outros planetas, a mineração sustentável de asteroides ou até mesmo a construção de constelações de satélites em órbita, é crucial a redução destes. Nesse contexto, a reutilização dos boosters emerge como a principal estratégia para atingir tais objetivos, não apenas reduzindo os gastos relativos ao veículo em até 42% (TOMANEK; HOSPODKA, 2018), mas também permitindo a redução da janela de tempo entre lançamentos.

Contudo, a reutilização de boosters é um campo de estudo ainda recente, marcado pelo êxito inaugural em dezembro de 2015, quando ocorreu o primeiro pouso bem-sucedido do primeiro estágio do foguete Falcon 9 da empresa norte-americana SpaceX (WALL, 2015). Por este motivo, juntamente com a sensibilidade do tópico nas questões de segurança e de sigilo industrial, carecem de estudos sobre controladores para tais missões.

Assim sendo, este trabalho propõe o desenvolvimento um sistema de controle para a etapa final de recuperação de um booster, conhecida como Landing Burn. Essa fase crítica do processo de reutilização, ilustrada esquematicamente na Figura 1, representa os últimos momentos de resposta do foguete antes de tocar o chão.

Explorou-se métodos de aprendizado de máquinas por reforço como uma alternativa aos tradicionais controladores PID (Proportional Integral Derivative), visando obter uma solução robusta, com boas respostas a ruídos, como eventuais turbulências ou perturbações não programadas.

Figura 1 – Perfil de missão de um Falcon 9



Fonte: Gardi e Ross (2019)

1.1 OBJETIVOS

Para resolver a problemática do pouso de boosters, propõem-se os seguintes objetivos.

1.1.1 Objetivo geral

Desenvolver um sistema de controle para a fase final de aproximação do booster ao solo utilizando métodos de apredizado de máquinas por reforço e comparar resultados simulados com dados de missões reais.

1.1.2 Objetivos Específicos

- Simular a dinâmica de um booster;
- Treinar modelos de controle sob diferentes condições de voo;
- Avaliar a eficácia dos modelos obtidos;
- Comparar as trajetórias simuladas com resultados de missões reais.

2 FUNDAMENTAÇÃO TEÓRICA

Visando obter uma metodologia eficaz para o desenvolvimento de controladores eficientes, altamente confiáveis e adaptáveis a diversos contextos, é essencial compreender conceitos fundamentais sobre controle, dinâmica e características do foguete e, principalmente, tópicos sobre aprendizado de máquinas por reforço.

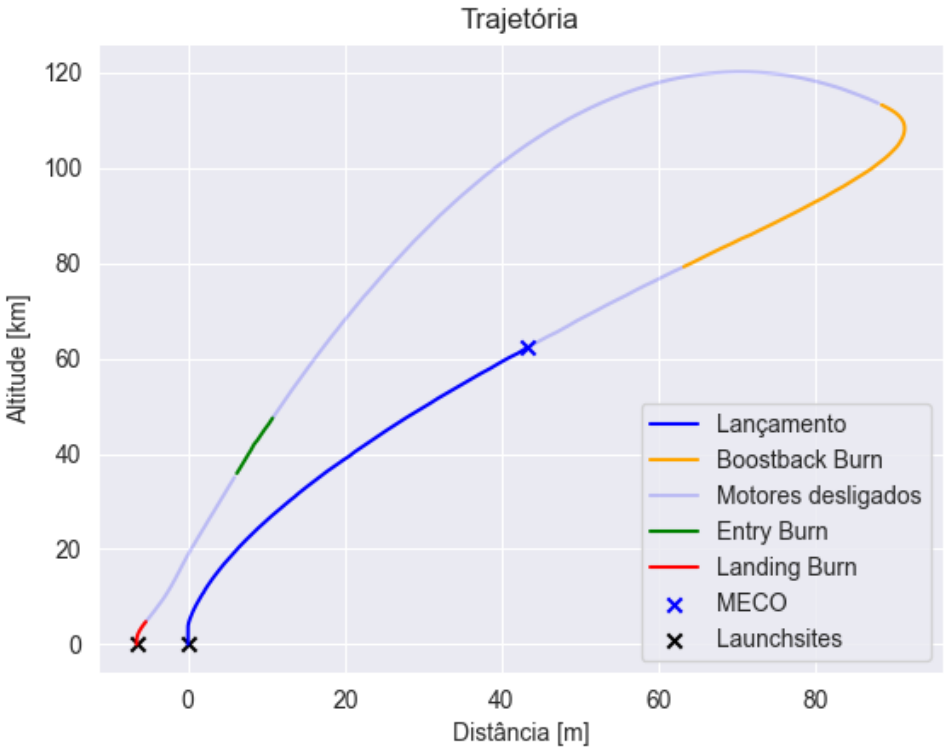
2.1 MISSÃO

A realização deste projeto baseia-se na missão CRS-11 da empresa SpaceX, cujo objetivo foi o reabastecimento da Estação Espacial Internacional (ISS) em junho de 2017. Os dados, disponibilizados por Shalev (2024), foram obtidos utilizando técnicas de visão computacional aplicadas aos dados de telemetria extraídos da transmissão oficial da empresa, seguidas de pós-processamento para detecção e correção de eventuais anomalias registradas, resultando na obtenção de métricas relevantes da missão.

São dois tipos de missão para a recuperação do primeiro estágio: Return To Launch Site (RTLS), que visa trazer o booster de volta à terra, em um local próximo ao ponto de lançamento, e a missão do tipo Autonomous Spaceport Drone Ship (ASDS), cujo destino final é uma balsa autônoma no oceano. Em geral, a segunda forma é mais complexa, pois envolve fatores como a instabilidade da balsa gerada pelas ondulações marítimas e a necessidade de coordenação precisa entre a posição da balsa e do veículo. No entanto, essa abordagem permite a recuperação em missões onde o combustível disponível na fase de retorno é inferior ao necessário para um RTLS.

Conforme ilustrado na Figura 2, a missão de referência configura-se como uma RTLS, composta pelas seguintes etapas: inicialmente, ocorre o lançamento até que os motores principais do primeiro estágio sejam desligados (MECO) e os estágios sejam desacoplados . Em seguida, realiza-se o *boostback burn*, onde os motores do booster são religados com o objetivo de redirecionar o foguete de volta ao local de lançamento. Posteriormente, a missão é conduzida pela velocidade residual, pela gravidade e pelos mecanismos de controle da aeronave, até o momento da *entry burn*. Nesse ponto, a atmosfera densa causa fenômenos característicos de regime hipersônico, como altas temperaturas e elevada pressão dinâmica nas estruturas, o que exige a religação dos motores para reduzir os estresses mecânicos. Finalmente, próximo ao local de lançamento, realiza-se a última queima conhecida como *landing burn*, durante a qual as velocidades e ângulos são ajustados até o momento do pouso, encerrando assim todas as operações.

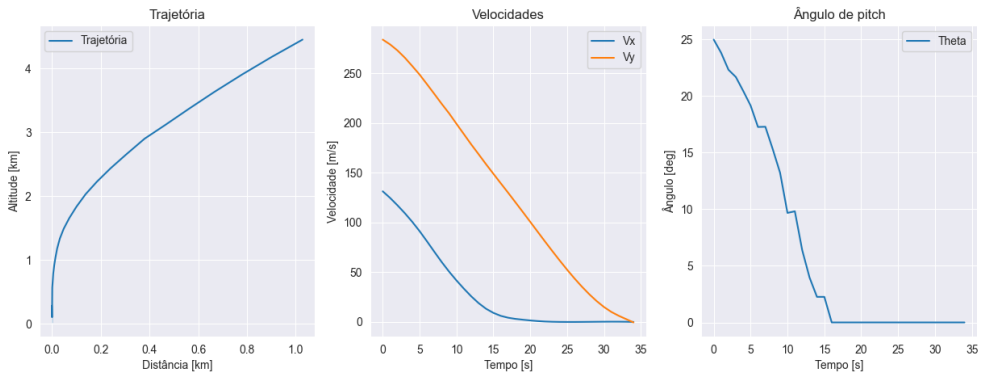
Figura 2 – Perfil da missão CRS-11



Fonte: Autor

Assim sendo, o objetivo principal deste trabalho é resolver o problema de controle da fase final de aproximação ao solo (Landing Burn), tendo em vista que as diferentes etapas possuem desafios e modelagens específicas, dificultando a solução por meio de um único controlador. Na Figura 3, ilustra-se a trajetória, velocidades e ângulo de pitch, que são os objetos de comparação nos resultados do trabalho.

Figura 3 – Métricas da fase de landing burn da missão CRS-11



Fonte: Autor

2.2 TEORIA DO CONTROLE

A teoria do controle é um campo multidisciplinar, que usa dos dados de sensores como entrada, para alcançar critérios de estabilidade e desempenho do projeto, através de respostas do sistema. Dentre as diversas aplicações, cita-se o controle de velocidade de veículos (cruise-control), sistemas de aquecimento, ventilação e ar-condicionado (HVAC), sistemas fly-by-wire em aeronaves e estabilizadores de foguetes (BRUNTON; KUTZ, 2019).

São diversos os modelos de controladores dentro da teoria, divididos em abordagens clássicas como controladores Proporcionais Integrais Derivativos (PID) e os baseados em técnicas de resposta em frequência, derivados do método Root Locus. No entanto, apresentam limitações que devem ser consideradas no seu uso.

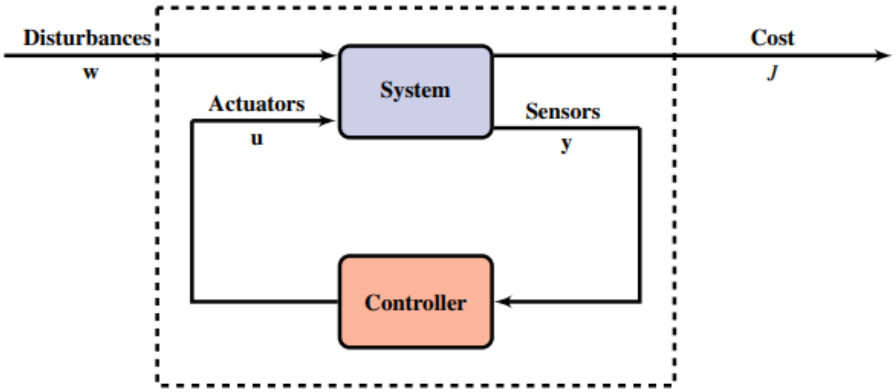
Primeiramente, a sintonização dos parâmetros PID pode ser um processo complexo e demorado, especialmente em sistemas não lineares ou com características dinâmicas variáveis. Além disso, controladores PID podem ter desempenho insatisfatório em sistemas com atrasos de tempo elevados, levando a oscilações e instabilidade. Outro ponto crítico é a sensibilidade a ruídos, onde a ação derivativa pode amplificar ruídos de alta frequência, prejudicando a resposta do sistema. Finalmente, em sistemas altamente interativos ou com múltiplas variáveis de controle, o PID pode ser insuficiente para atender a todas as necessidades de controle, exigindo abordagens mais avançadas ou complementares (FRANKLIN et al., 2019).

Por essas razões, surgem abordagens modernas, viabilizadas pelos avanços na capacidade computacional dos sistemas embarcados, como os controladores de modelo preditivo (MPC) e controladores que fazem uso de técnicas de inteligência artificial, como redes neurais artificiais e algoritmos de aprendizado de máquina. Essas técnicas modernas destacam-se devido a sua capacidade de lidar com sistemas complexos e não lineares, ajustando-se dinamicamente às condições do sistema em tempo real e respondendo a ruídos sem desestabilizar o sistema.

2.2.1 Sistemas dinâmicos

Um sistema dinâmico é entendido como uma representação matemática que descreve o comportamento evolutivo de um processo específico a ser controlado no tempo. Este consiste em um conjunto de equações diferenciais, acopladas ou não, cujo objetivo é descrever as variáveis de interesse ao longo do tempo e suas respostas a perturbações. Tais modelos dinâmicos são derivados predominantemente a partir de equações físicas fundamentais ou, alternativamente, através da análise empírica das respostas observadas de um sistema a estímulos externos controlados (FRANKLIN et al., 2019).

Figura 4 – Esquema de feedback de loop fechado



Fonte: Brunton e Kutz (2019)

2.2.2 Feedback de loop fechado

Tratando-se de sistemas com incertezas, instabilidades e perturbações externas, é comum utilizar sistemas de feedback de loop fechados, como ilustrado na Figura 4. Nela, medições dos sensores y são passadas como entrada ao controlador, junto com as incertezas do sistema w, retornando um sinal u para os atuadores modificarem a dinâmica do sistema (BRUNTON; KUTZ, 2019).

No esquema acima, decompõe-se o vetor de incertezas como $w = [w_d^T w_n^T w_r^T]^T$, onde w_d são as perturbações do sistema, como uma rajada de vento. w_n o ruído das medições dos sensores e w_r uma trajetória de referência do sistema que deve ser medida pelo sistema de loop fechado.

Matematicamente, a dinâmica do sistema e as medições podem ser modeladas segundo as equações a seguir, onde x representa o estado do sistema em um dado instante de tempo,

$$\frac{\mathrm{d}x}{\mathrm{d}t} = f(x, u, w_d),$$
$$y = g(x, u, w_n).$$

O objetivo do problema é encontrar uma lei de controle u, tal que:

$$u = k(y, w_r),$$

que minimize uma função custo J, onde:

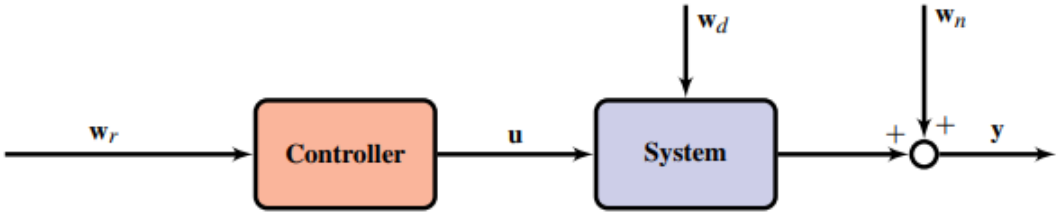
$$J = J(x, u, w_r),$$

e, portanto, dependendo de tecnicas de otimização numéricas para isso.

Para efeitos de comparação, vale citar os sistemas de feedback de loop aberto, conforme ilustrados no esquema da figura 5. Nestes, não são incorporados ruídos e

perturbações do sistema na tomada de decisão do controlador, resultando na incapacidade deste tipo de modelo para estabilizar sistemas instáveis. Dado um sistema para controle de velocidade automobilístico (cruise-control), é simulada a resposta de ambos modelos ao ruído induzido ao sistema na tentativa de estabilizar a velocidade em um valor fixo, na figura 6.

Figura 5 – Esquema de feedback de loop aberto



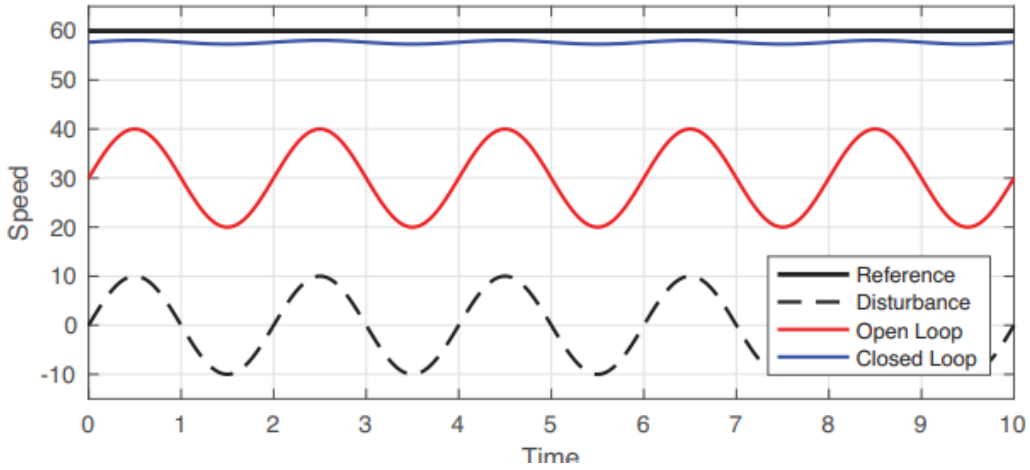
Fonte: Brunton e Kutz (2019)

2.3 APRENDIZADO POR REFORÇO PROFUNDO

O aprendizado por reforço (RL) pode ser entendido como a abordagem que formaliza, de maneira matemática e computacional, o aprendizado de seres vivos observado na vida real. Nesse sentido, as interações entre um agente e seu ambiente são refletidas na forma de recompensas do sistema dopaminológico, isto é, ações que gerem estímulos positivos devem ser reforçadas ao passo que o contrário, desistimulado.

Assim como qualquer área relacionada a Machine Learning (ML), os modelos não são explicitamente instruídos sobre como agir frente as diferentes situações,

Figura 6 – Comparação entre modelos de feedback



Fonte: Brunton e Kutz (2019)

mas possuem a capacidade de aprender com os dados. Contudo, ao contrário de abordagens clássicas, como aprendizado supervisionado e não supervisionado, onde os dados são conhecidos com antecedência, em RL estes são obtidos por meio de inúmeras experiências, positivas ou negativas, de interação entre agente e ambiente. Aqui, o objetivo é maximizar recompensas numéricas acumuladas ao longo do tempo, tendo o estado atual como entrada e a resposta do ambiente ao agente como saída (BARTO; ROSS, 2018).

Portanto, dentro do aprendizado por reforço profundo alguns pontos são de suma importância e serão frequentemente citados neste trabalho, abordados brevemente na sequência.

2.3.1 Redes Neurais Artificiais

Inspirado no funcionamento dos neurônios biológicos e a forma como estes se organizam e interagem entre sí, assim como em trabalhos anteriores de outros autores, Fukushima (1980) propôs em seu trabalho "A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position", uma representação matemática que mimetizava, em muitas formas, o funcionamento do cerébro para tomada de decisões e interpretação de padrões. Contudo, foi somente nas útimas décadas, impulsionado pelos avanços na capacidade computacional, pela disponibilidade de grandes volumes de dados rotulados e por progressos teóricos significativos, que o potencial das redes neurais artificiais foi plenamente realizado (LECUN et al., 2015).

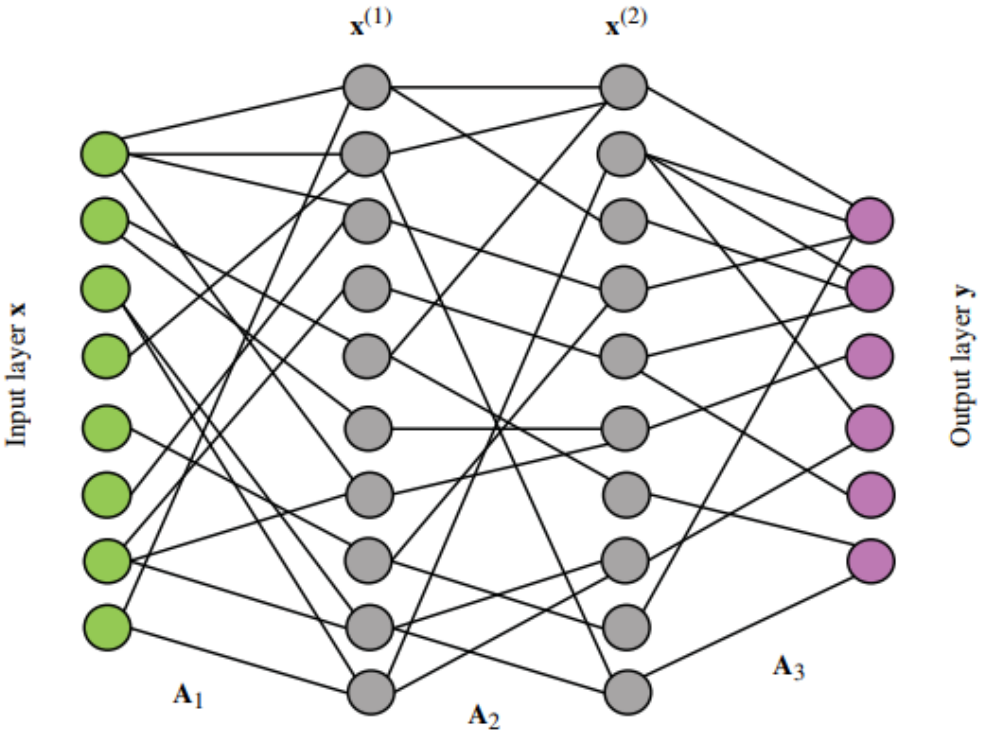
Atualmente, essas redes são aplicadas em uma variedade de contextos, incluindo reconhecimento e geração de imagens, detecção de fraudes financeiras, desenvolvimento de controladores, entre outras, beneficiando-se da capacidade desse tipo de arquitetura em identificar padrões intrínsecos nos dados, muitas vezes imperceptíveis aos observadores humanos.

Essas redes podem ser modeladas de diversas maneiras para atender a diferentes requisitos e resolver uma ampla gama de problemas. Por exemplo, as Convolutional Neural Networks (CNN) são empregadas em situações onde os dados são predominantemente imagens; as Recursive Neural Networks (RNN), por sua vez, são adequadas para dados com uma estrutura temporal a ser considerada; e as Feedforward Neural Networks (FNN), representadas na Figura 7, são consideradas a arquitetura padrão para estimadores (HORNIK et al., 1989).

Uma camada qualquer de uma FNN, pode ser modelada matematicamente segundo a equação 1,

$$h = \sigma(\boldsymbol{x} \cdot \boldsymbol{W} + b), \tag{1}$$

Figura 7 – Feedfoward Neural Network (FNN)



Fonte: Brunton e Kutz (2019)

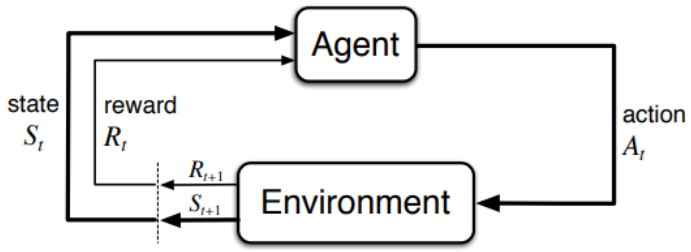
em que o produto vetorial entre $\boldsymbol{x}$, um vetor de valores de entrada, e $\boldsymbol{W}$, os pesos de cada neurônio da camada, somados de b, um fator de viés, são transformados através da função de ativação σ, resultando nos valores de saída h. Esta, a função de ativação, tem como objetivo introduzir a não linearidade nas respostas da rede, a principal diferença entre arquiteturas de redes neurais e modelos de regressão lineares tradicionais dentro do aprendizado de máquinas, possibilitando a essas a capacidade de generalizar padrões complexos.

Portando, a arquitetura da rede se torna, nesta sequência, uma camada de entrada $\boldsymbol{X}$, cujos valores são conhecidos como features. Seguido por camadas "escondidas", cujos vetores de pesos e viéses $\boldsymbol{W}$ e b são aprendidos durante o treinamento da rede. O número de camadas e a quantidade de neurônios (ou nós) da rede, é uma hiperparâmetro a ser ajustado ao otimizar o modelo, enquanto o tamanho das camadas de entrada e saída Y são definidos segundo o problema de interesse.

2.3.2 Interação ambiente-agente

Segundo Barto e Ross (2018), um processo de decisão de Markov (MDP) é a formalização de um processo de decisão sequencial, onde ações influenciam não só o próximo instante, mas também os posteriores. É também definido que a probabilidade

Figura 8 – Interação agente-ambiente em um processo de decisão de Markov



Fonte: Barto e Ross (2018)

de se encontrar em um estado futuro é determinada apenas pelo estado atual, não por estados anteriores ou fatores externos aos conhecidos. Esta é a forma matemática idealizada do problema de aprendizado por reforço, na qual as hipóteses do problema e solução serão baseadas.

Assim sendo, assume-se o *ambiente* de um problema de RL como um MDP, que consiste em um conjunto de *estados* $\mathcal{S}$, de ações $\mathcal{A}$ e recompensas $\mathcal{R}$. Neste, a probabilidade de transição de um estado s_t para outro s_{t+1}, dada uma ação a_t é,

$$P(s', s, a) = \Pr(s_{t+1} = s' | s_t = s, a_t = a).$$

Em síntese, conforme ilustrado na figura 8, o agente avalia seu estado $s_t \in \mathcal{S}$, toma uma ação $a_t \in \mathcal{A}$ de tal maneira a maximizar a recompensa recebida $r_t \in \mathcal{R}$, dada pelo ambiente após a transição para s_{t+1}.

2.3.3 Recompensa

Para o agente, o objetivo de seu aprendizado é retratado como um sinal numérico, a *recompensa*. Esta, dada pelo ambiente a cada instante, pode ser entendida como um feedback de sucesso que, acumulado ao longo do tempo, deve ser maximizado em busca da solução ideal do problema.

Formaliza-se a recompensa da transição entre estados em um MDP, segundo

$$\mathbf{R}_t(s', s, a) = \Pr(r_{t+1} | s_{t+1} = s', s_t = s, a_t = a).$$

e o retorno como,

$$\mathbf{G}_t = \mathbf{R}_{t+1} + \mathbf{R}_{t+2} + \mathbf{R}_{t+3} + \cdots + \mathbf{R}_T, \tag{2}$$

onde T é o passo de tempo final, marcando o final de um *episódio*, o conjunto de todas as transições de estado até s_T, conhecido como estado terminal. Este pode informar ao agente a forma como o episódio acabou: sucesso, falha ou algum estado intermediário (BARTO; ROSS, 2018).

Contudo, em muitos problemas o passo de tempo final T não é conhecido, ou $T \to \infty$ e, portanto, o retorno também será desconhecido. Este é conhecido como problema da diferença temporal e é abordado na sequência.

2.3.3.1 Problema da Diferença Temporal

Conforme já abordado, o objeto de interesse de problemas de RL é a maximização do retorno que o agente obterá. Contudo, a indefinicão deste valor inviabiliza, ou ao menos dificulta significativamente, o aprendizado.

Visando solucionar tal questão, introduz-se a *taxa de desconto* das recompensas γ, e o retorno descontado de um episódio é dado pela equação 3,

$$\mathbf{G}_t = \mathbf{R}_{t+1} + \gamma\mathbf{R}_{t+2} + \gamma^2\mathbf{R}_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k \mathbf{R}_{t+k+1}\,, \quad 0 \leqslant \gamma \leqslant 1 \tag{3}$$

Segundo esta, à medida que γ tende a 1 o agente passa a considerar os efeitos de suas ações a longo prazo mais firmemente. Na contravia, tendendo a 0 as consequências imediatas são priorizadas. Em outras palavras, o fator de desconto determina o valor presente das recompensas futuras (BARTO; ROSS, 2018).

2.3.4 Política, Valor e Qualidade

Em geral, o objetivo dos algorítimos de RL envolve a estimativa da *função valor* do problema, que procura determinar o quão bom é para o agente estar em determinado estado s. Alternativamente, pode estimar a qualidade de se tomar uma ação a quando em um estado s. Tal estimativa se traduz como a recompensa futura esperada $\mathbf{G}_t$, consequência essa das ações tomadas pela *política* (BARTO; ROSS, 2018).

A função valor $v_\pi(s)$ de um estado s, sob uma política qualquer π, em um MDP, formula-se segundo a equação 4,

$$v_\pi(s) = \mathbb{E}(\sum_{k=0}^{\infty} \gamma^k r_k | s_0 = s), \tag{4}$$

também conhecida como função *valor-estado da política* π.

Por sua vez, a política $\pi(a|s)$, é definida como a função que mapea estados $s_t \in \mathcal{S}$ em *probabilidades* de tomar cada possível ação $a_t \in \mathcal{A}$. Portanto, métodos de RL especificam como a política deve se alterar como resultado de sua experiência (BARTO; ROSS, 2018). A política pode ser deterministica,

$$a_t = \mu(s_t),$$

quando o mesmo estado s sempre retorna a mesma ação a, denotada como $\mu(s)$, ou estocástica,

$$a_t \sim \pi(\cdot|s_t),$$

onde as ações são retornadas na forma de uma distribuição probabilistica. Na prática, utiliza-se a denotação para a política como π em ambos os casos.

Partindo do princípio que é conhecida a melhor política possível, também chamada de política ótima π_*, escreve-se a função valor segundo a equação 5,

$$v(s) = \max_{\pi} \mathbb{E}(\sum_{k=0}^{\infty} \gamma^k r_k | s_0 = s), \tag{5}$$

e ao utilizar a propriedade da recursividade da equação, encontra-se a equação 6,

$$v(s) = \max_{\pi} \mathbb{E}(r_0 + \gamma v(s')), \tag{6}$$

onde $s' = s_{k+1}$ é o estado sequente a $s = s_k$ dado a ação a_k selecionada pela política π. Essa é conhecida como equação de Bellman (BELLMAN, 1957), referente ao princípio da otimalidade que norteia as abordagens modernas na área (BRUNTON; KUTZ, 2019). Consequentemente, extrai-se dessa a política ótima citada anteriormente, na equação 7,

$$\pi_*(s) = \arg\max_{\pi} \mathbb{E}(r_0 + \gamma v(s^{'})). \tag{7}$$

Por fim, é comum a utilização de outra formulação para a função valor parametrizada para os pares estado-ação, ao invés de apenas o estado, chamada de função qualidade Q, como na equação 8,

$$Q(s, a) = \mathbb{E}(R(s', s, a) + \gamma v(s')), \tag{8}$$

cuja relação com a política e valor são dadas nas equações 9 e 10, respectivamente.

$$\pi(s, a) = \arg\max_{a} Q(s, a), \tag{9}$$

$$v(s) = \max_{a} Q(s, a). \tag{10}$$

2.3.4.1 Deep RL

Em problemas simples, é possível modelar funções capazes de mapear os espaços discretos S para $\mathbf{A}$, em que a solução se dará na forma tabular. No entanto, esta abordagem não é eficiente, ou até mesmo impossível, para a grande maioria dos casos. Em geral, a política é representada por uma aproximação parametrizada por um vetor de dimensão inferior θ, em que:

$$\pi(s, a) \approx \pi(s, a, \theta),$$

comumente denotado apenas como $\pi_\theta(s, a)$ (BRUNTON; KUTZ, 2019).

Essa é a base na qual a política, função valor e outras funções são construídas quando trata-se de Deep Reinforcement Learning (Deep RL), em que redes neurais são utilizadas como alternativa as tabelas anteriormente citadas. Portanto, políticas passam a ser representadas como $\pi_\theta(\cdot|s_t)$ e $\mu_\phi(s_t)$, onde ϕ e θ são um conjunto de viéses e pesos aprendidos durante o treinamento (OPENAI, 2018a).

3 METODOLOGIA

Nesta seção são apresentadas as ferramentas, hipóteses e modelos utilizados na modelagem do ambiente de simulação. Assim como o algorítimo de aprendizado por reforço utilizado para a solução do problema.

3.1 DINÂMICA DA SIMULAÇÃO

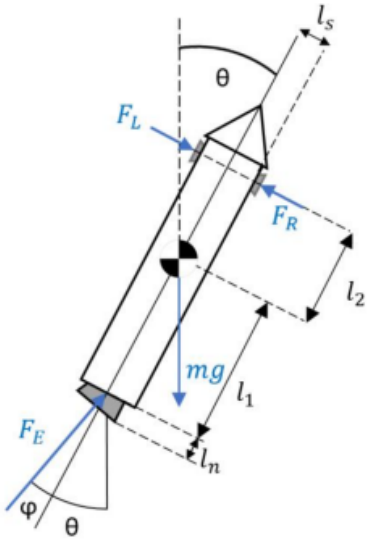
A partir do diagrama ilustrado na figura 9, são deduzidas as equações que regem o sistema dinâmico de interesse. Todas as dimensões foram baseadas para representarem o primeiro estágio de um Falcon 9, segundo SpaceX (2021). Contudo, devido a limitações na disponibilidade de informações, algumas métricas e hipóteses provem de estimativas de mecanismos similares.

O sistema dinâmico de interesse é descrito conforme a Equação 11,

$$\begin{bmatrix} \dot{m} \\ \dot{x} \\ \dot{y} \\ \dot{\theta} \\ \dot{v}_x \\ \dot{v}_y \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 \\ v_x \\ v_h \\ \omega \\ \frac{F_x}{m} \\ \frac{1}{m}[F_y + W] \\ \frac{1}{I}[-\dot{m}\omega r_m^2 + M] \end{bmatrix} + \begin{bmatrix} f_m(F_M) \\ 0 \\ 0 \\ 0 \\ \frac{1}{m}[F_M \cos(\delta + \theta) + F_s \cos\theta] \\ \frac{1}{m}[F_M \sin(\delta + \theta) + F_s \sin\theta] \\ \frac{1}{I}[l_m T_M \sin\delta + l_s T_s] \end{bmatrix} \tag{11}$$

onde $\dot{(\,)}$ corresponde à derivada no tempo, f_m é uma função para estimar a variação da massa de propelente conforme a potência do motor, l_m corresponde a distância

Figura 9 – Diagrama de forças do booster



Fonte: Ferrante (2017)

longitudinal entre o centro de massa do bocal até o centro de massa do booster, enquanto l_s às distâncias dos motores laterais à linha de centro vertical. F_y, F_x e M são eventuais componentes de forças e torques provenientes dos modelos de vento e turbulência que atuam sob o booster. F_m e F_s são as forças aplicadas pelo motor principal e sistemas de controle, respectivamente.

3.2 AMBIENTE DE SIMULAÇÃO

Buscando modelar um ambiente conforme um MDP, segundo recomendado pela literatura, utiliza-se a biblioteca *gymnasium* para *Python* da empresa *OpenAI*. Essa disponibiliza um API de alto nível de abstração de um ambiente de aprendizado de máquinas por reforço, sendo a base na qual o ambiente será construído.

Para simular a dinâmica do booster, optou-se pela utilização da engine física de colisão de objetos rígidos Box2D, devido à sua facilidade de implementação. Esta engine apresenta três graus de liberdade, permitindo a translação e rotação no plano xy. Conforme indicado na documentação, são utilizados métodos de primeira ordem na implementação, os quais, embora não sejam ideais, são suficientes para o contexto deste trabalho. Indica-se também trabalhar com dimensões entre 0,1 a 10 metros, devido a possíveis erros de memória durante as colisões (CATTO, 2023). Por essa razão, as dimensões físicas reais passam a ser representadas em uma escala 1:100, posteriormente convertidas para valores nominais após a simulação. As seguintes variáveis, na tabela 1, também sofrem correção após validações empíricas para condizerem com a realidade.

Tabela 1 – Constantes de correção do ambiente.

Variável	Constante
Tempo	10
Velocidade linear	10
Velocidade ângular	0.04293

Fonte: Autor

3.2.1 Observação

As informações disponíveis na observação do ambiente são modeladas para corresponderem a métricas disponíveis pelos sistemas embarcados da aeronave, que são utilizadas como entrada para o controlador de interesse. Assim, o vetor estado é $s = \begin{bmatrix} \Delta x & \Delta y & V_x & V_y & \theta & \omega & \phi \end{bmatrix}$, cujas variáveis são descritas na tabela 2, juntamente com os limites aplicáveis.

Em paralelo, algumas das variáveis estão sujeitas a restrições que, quando excedidas, resultam em terminações dos episódios. Essas são divididas em duas

Tabela 2 – Variáveis de estado do ambiente

Variável	Descrição	Limite
x [m]	x_{CG} em relação a x_{target}	$[-\infty, +\infty]$
y [m]	y_{CG} em relação a y_{target}	$[-\infty, +\infty]$
θ [°]	Ângulo de pitch	$[-\infty, +\infty]$
V_x [m/s]	Velocidade linear x	$[-\infty, +\infty]$
V_y [m/s]	Velocidade linear y	$[-\infty, +\infty]$
ω [rad/s]	Velocidade angular	$[-\infty, +\infty]$
φ [°]	Ângulo de vetorização do bocal	[-12,+12]
ϕ	Fração de combustível restante	[0,1]

Fonte: Autor

categorias, a primeira descreve as condições de estabilidade durante o voo, onde supõem-se que após atingidos os limites torna-se uma condição irrecuperável. A segunda, por sua vez, refere-se as condições que definem um pouso bem sucedido, isto é, se as variáveis forem superiores aos limites impostos no momento em que o booster toca o chão, considera-se uma explosão. Tais condições são descritas na tabela 3, a seguir.

Tabela 3 – Variáveis de estado do ambiente

Variável	Limite de voo	Limite de pouso
x [m]	$[-\infty, +\infty]$	Raio do launchpad
θ [°]	$[-60, +60]$	[-23,+23]
V_x [m/s]	$[-\infty, +\infty]$	[-3, +3]
V_y [m/s]	$[-\infty, +\infty]$	[-10, 0]

Fonte: Autor

3.2.2 Ação

Por sua vez, o controlador emite em todos os instântes uma resposta $a = \begin{bmatrix} F_m & F_s & \Delta\varphi \end{bmatrix}$, todos limitados por [-1,+1]. Assim, o empuxo empregado pelo motor principal F_m varia de 57% a 100% da potência máxima disponível, proporcionalmente, quando F_m é superior a 0, caso contrário o motor encontra-se desligado.

Para os motores laterais, ativa-se o thruster esquerdo quando F_s está compreendido entre [-1,-0.5] e o direito [0.5,1]. Finalmente, o ângulo de vetorização do bocal varia segundo $\varphi = \Delta\varphi\Omega\Delta t$, onde Ω é a velocidade angular empregada pelos atuadores do bocal.

3.2.3 Recompensas

A recompensa do ambiente será modelada utilizando a técnica de *Potential based reward shaping*, conforme descrito por Russell Daishi Harada (1999). Nessa, a função recompensa R do MDP passa a possuir um fator de shaping F, representado por

$$F(s, s') = \gamma\Phi(s') - \Phi(s),$$

onde γ é o fator de desconto das recompensas e Φ representa o potencial de um estado s qualquer, conforme a Equação 12,

$$\Phi(s) = \lambda\eta, \tag{12}$$

em que λ é um vetor de constantes que balanceará a importancia de cada um dos elementos de η, variáveis de interesse para o objetivo final.

Nessa abordagem, o fator de shaping F será negativo caso a transição entre estados seja contra a trajetória esperada, caso contrário será positiva. Portanto, neste trabalho define-se

$$\lambda = \begin{bmatrix} R_v & R_s & R_\theta \end{bmatrix},$$

como as constantes de recompensa da velocidade, posição e ângulo, respectivamente. Consequentemente, o vetor de variáveis será

$$\eta = \begin{bmatrix} ||V|| & S & \theta \end{bmatrix}^T,$$

referentes a magnitude do vetor velocidade V, da distância S do booster em relação ao launchpad e ao valor absoluto do ângulo do booster θ.

Em conjunto, a recompensa do ambiente incorpora punições para incentivar comportamentos específicos, representado por uma função $P(a, a', s, s')$, que considera busca atingir os comportamentos descritos na tabela 4.

Tabela 4 – Ações de controle do booster

Gatilho	Comportamento esperado
Ignição do motor principal	Uma única queima
Potência do motor principal	Economia de propelente
Ignição dos motores laterais	Economia de propelente e estabilidade
Variação do ângulo de vetorização	Estabilidade

Fonte: Autor

Finalmente, modela-se a recompensa de terminação segundo a Equação 13, a seguir,

$$r_{ter} = R_{ter} + \lambda\eta_{ter} \tag{13}$$

onde η_{ter} é o conjunto de constantes de recompensa terminal, equanto R_{ter} é uma constante positiva em caso de um pouso bem sucedido e negativa nos casos de explosão ou instabilidade, quando alguma das variáveis excede os limites máximos permitidos.

Portanto, a recompensa do ambiente é dada segundo,

$$R(a, a', s, s') = \begin{cases} F(s, s') + P(a, a', s, s') & \text{se } s \neq s_{ter} \\ r_{ter} & \text{se } s = s_{ter} \end{cases} \tag{14}$$

3.2.4 Booster

3.2.4.1 Propulsão

Utiliza-se o software CEARUN, desenvolvido pela National Aeronautics and Space Administration (NASA), capaz de estimar condições de saída de propriedades relevantes em um motor foguete, a partir de hipóteses de escoamento congelado e de equilíbrio, sendo a primeira a selecionada para a modelagem.

Considera-se que a velocidade de saída do escoamento após a câmara de combustão é significativamente maior que a velocidade de reação das espécies químicas envolvidas, por isso a composição química do escoamento se mantém constante durante a expansão no bocal e as propriedades aerodinâmicas variam (ANDERSON, 2006).

Modela-se a propulsão do booster a partir de um motor Merlin 1D, da empresa SpaceX. Esses são propulsores *full-flow* capazes de produzir 854 kN de empuxo ao nível do mar, alimentados por RP-1 e oxigênio líquido (LOX), a uma razão de mistura ϕ=2,36 e coeficiente de expansão (razão entre área na garganta e na saída do bocal) igual a $\varepsilon = 21$.

Finalmente, com esses dados como entrada para a simulação, determina-se a velocidade e pressão de saída estimadas, dado empuxo máximo, de 803 m/s e 65,4 kPa respectivamente. Assim, é possível estimar o consumo de massa de propelentes durante o voo a partir da Equação 15,

$$F_{main} = \dot{m} V_e + (P_e - P_a) A_e \tag{15}$$

onde V_e e P_e representam a velocidade e pressão de saída do escoamento, enquanto P_a a pressão ambiente e A_e a área de saída do bocal. Considerando as hipóteses simplificadoras de pressão ambiente como a do nível do mar, pressão de saída e velocidade de escape dos gáses constantes durante toda a missão, é encontrada a Equação 16 do fluxo de massa em função do empuxo empregado por um dos motores, sob a razão de mistura ϕ,

$$\dot{m}(F_{main}) = \frac{F_{main} - (P_e - P_a) A_e}{V_e}. \tag{16}$$

Figura 10 – Propulsores de um Falcon 9

Fonte: SpaceX (2021)

Um booster Falcon 9 possui 9 motores distribuídos segundo a imagem 10, todos com capacidade de vetorização, cujo grau não está disponível pela empresa e por isso estima-se 10° de liberdade. Possuem também controle de potência, com um mínimo de 57% da capacidade máxima. Contudo, durante as fases de recuperação do veículo lançador utiliza-se 1 ou 3 motores apenas, a depender da configuração da missão.

3.2.4.2 Controle de atitude

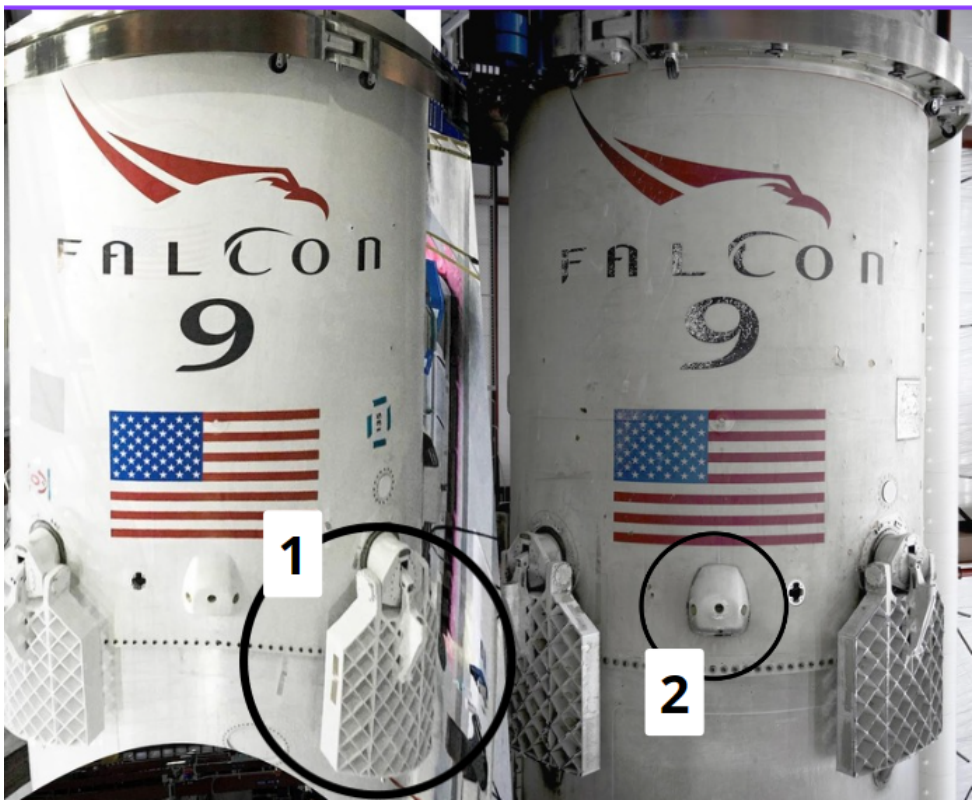
A fim de auxiliar no controle e estabilidade da aeronave ao longo das fases de retorno, são empregados dois sistemas auxiliares afastados do centro de gravidade da aeronave, ilustrados na figura 11.

As *grid fins*, no círculo 1, são superfícies aerodinâmicas que direcionam o escoamento em determinadas direções, gerando forças conforme necessário, ao ajustar o ângulo em dois graus de liberdade de rotação. Essas são proporcionais a velocidade da aeronave, assim como a viscosidade dinâmica do ar e por isso são ineficientes nos estágios iniciais, em que a densidade do escoamento é baixa em grandes altitudes, e finais da missão de recuperação, em que as velocidades não se fazem suficientes.

Paralelamente, no círculo 2, existem propulsores a gás, geralmente alimentados com nitrogênio, capazes de gerar esforços a partir dos gases expelidos em alta velocidades. Esses podem e são utilizados em qualquer etapa da missão de retorno para correções no ângulo de ataque do veículo.

Contudo, é difícil modelar as eficiências dos sistemas devido a complexidades

Figura 11 – Sistemas de controle do Falcon 9



Fonte: SpaceX

nos processos aerodinâmicos envolvidos nos *grid fins*, ou a falta de informação relativa as pressões dos sistemas de propulsores a gás para estimar as forças resultantes. Portanto, ambos sistemas são modelados como um único contando com 25 kN de força disponíveis, ajustados e disparados conforme necessário.

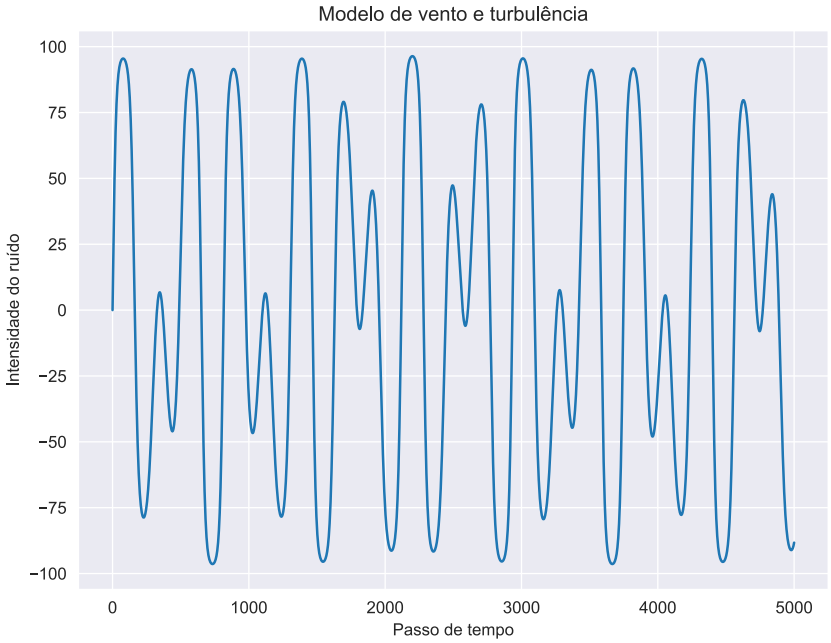
3.2.5 Vento e Turbulência

Um aspecto significativo no contexto da aprendizagem de máquina por reforço, e particularmente em problemas de controle, reside na integração de ruído aleatório nos dados com o intuito de mitigar o fenômeno de *overfitting* do modelo. Isto é, em vez de encontrar padrões intrínsecos dos dados, este passa a memorizar o conjunto de treinamento e, consequentemente, apresentar desempenho insatisfatório quando confrontado com situações inéditas (PLAPPERT et al., 2017).

Adicionalmente, do ponto de vista da teoria do controle, a presença de ruídos e incertezas é inevitável, podendo induzir perturbações irreversíveis na dinâmica do sistema. Contudo, controladores baseados em RL possuem alta capacidade de mitigar ruídos, sendo esse um dos principais motivadores dos estudos sobre aplicações de RL para problemas de controle (MOOS et al., 2022).

Com isso em mente, introduzem-se ao ambiente duas espécies de ruído aleatório: vento e turbulência. A primeira, aplica forças horizontais ao centro de

Figura 12 – Modelo de vento e turbulência do ambiente



Fonte: Autor

gravidade do booster, resultando em alterações nas variáveis de velocidade linear. A segunda, introduz torque, modificando assim a velocidade angular. Ambos os ruídos são descritos pela Equação 17, uma função não periódica segundo Farama (2024),

$$I = \tanh\{\sin[2k(t+C)] + \sin[\pi k(t+C)]\} * I_{max} \tag{17}$$

Ao início de cada episódio, uma número inteiro C é aleatóriamente selecionado, servindo de ponto de entrada na função de ruído. Em contrapartida, t é o passo de tempo atual da simulação. I_{max} e k são hiperparâmetros a serem definidos para definir a amplitude e freqûencia das oscilações, respectivamente. Uma amostra do modelo é representada na figura 12.

Portanto, a força do vento aplicada sobre o booster é,

$$F_{vento} = I_{vento}(t) * 1\,\text{N}, \tag{18}$$

e o momento gerado pela turbulência,

$$M_{turbulencia} = I_{turbulencia}(t) * 1\,\text{Nm} \tag{19}$$

É feita também a hipótese de que a presença de vento e ruído é pequena quando próximos ao nível do mar e, portanto, deixam de ser aplicadas ao veículo quando se encontra em altitudes inferiores a H_{ruido}, um hiperparâmetro a ser definido.

3.3 PROXIMAL POLICY OPTIMIZATION

No artigo Proximal Policy Optimization Algorithms, de Schulman et al. (2017), o autor propõe uma nova classe de algorítimos de RL baseados em métodos de policy gradient, que utilizam técnicas clássicas de gradiente descendente para realizar a otimização dos parâmetros das políticas de interesse.

A partir da política atual, o agente interage com o ambiente para coletar dados de como o comportamento definido impacta nas recompensas recebidas durante as trajetórias vistas, limitadas a um hiperparâmetro de treinamento. Em seguida, é realizado um passo de otimização da função objetivo "substituta"e da função valor, usando gradiente ascendente estocástico para minimizar o erro.

Durante esse passo, utiliza-se de múltiplas épocas de treinamento separadas em minibatches de dados, possibilitando uma melhor utilização dos dados coletados, melhorando a eficiência e a estabilidade do treinamento. Além disso, é adaptado conceitos do algoritmo Trust Region Policy Optimization (TRPO) para garantir estabilidade e boa convergência, sem as complexidades computacionais associadas ao TRPO. O algorítimo faz isso impondo penalidades sobre mudanças bruscas na política, em vez das restrição complexas do TRPO, simplificando a implementação ainda que mantendo a robustez. O algorítimo segue até que algum critério de parada, como o número máximo de atualizações dos pesos das redes, seja atingido.

São intercaladas as etapas de interação do agente com o ambiente para a obtenção de dados, utilizando a política atual, com a otimização de uma função objetivo "substituta"usando gradiente ascendente estocástico. Em paralelo, contrastando com os algorítimos da classe, traz multiplas épocas em *minibatches*. Finalmente, conceitos do algorítimo Trust Region Policy Optimization (TRPO) são adaptados para aproveitar a capacidade de estabilidade e convergência que esse possui.

O funcionamento proposto é brevemente explicado na forma de pseudocódigo em sequência, no algorítimo 1 (OPENAI, 2018b).

Na figura 13, comparam-se as recompensas médias, no eixo y, de treinamento para diversos ambientes *gymnasium*, todos treinados até 1 milhão de passos, entre algorítimos já validados dentro do aprendizado de máquinas por reforço. Comprova-se que o algorítmo proposto PPO supera em praticamente todos os casos a performance e velocidade de convergência em relação aos demais.

Em adição aos resultados positivos, trata-se de um algorítimo *robusto*, isto é, na maioria dos casos não é necessário procurar e ajustar hiperparâmetros para se obter desempenho ótimo, ao contrário dos algorítimos comparados e da maioria de RL como um todo. Por essas razões, é dito que o Proximal Policy Optmization é atualmente o *estado da arte* quando trata-se de RL. Em outras palavras, o algorítimo é a escolha certa para grande maioria dos problemas a serem resolvidos, com um ótimo equilibrio entre eficiência computacional e resultados obtidos. Portanto, partindo das justificativas

Algorithm 1: PPO

Input: Parâmetros iniciais da política θ_0, parâmetros iniciais da função valor ϕ_0

for $k = 0, 1, 2, \ldots$ **do**

Colete o conjunto de episódios $\mathcal{D}_k = \{\tau_i\}$ dada a política $\pi_k = \pi(\theta_k)$ no ambiente;

Calcule as recompensas acumuladas $\hat{R}_t$;

Calcule as estimativas de vantagem, $\hat{A}_t$ (usando qualquer método de estimação de vantagem) baseado na função de valor atual V_{ϕ_k};

Atualize a política maximizando o objetivo PPO-Clip:

$$\theta_{k+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right)$$

tipicamente via ascensão de gradiente estocástico com Adam;

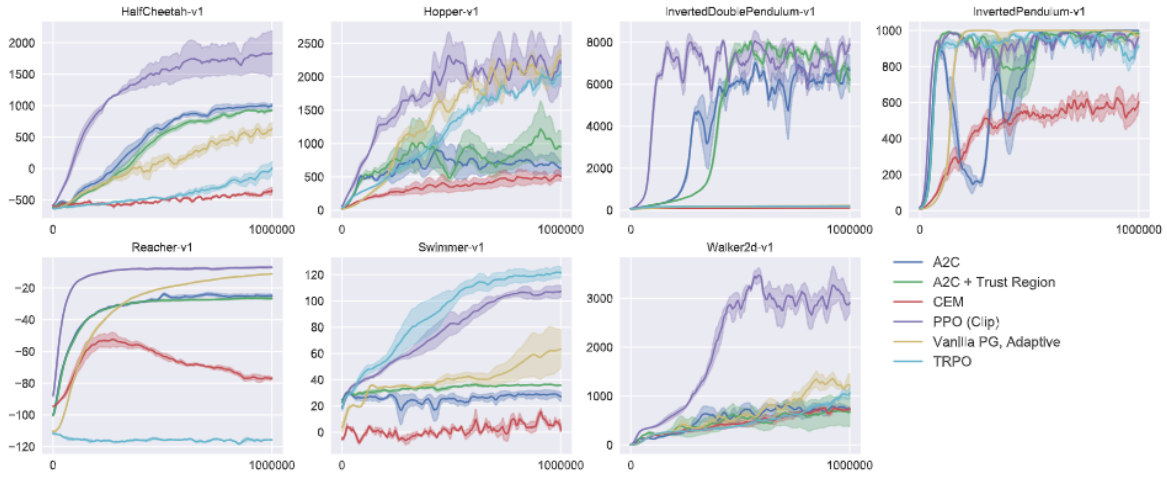
Ajuste a função de valor via regressão baseado no MSE:

$$\phi_{k+1} = \arg\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2$$

via algum algoritmo de gradiente descendente;

end

Figura 13 – Comparação entre algorítimos para diversos ambientes distíntos.



Fonte: Schulman et al. (2017)

explicitadas anteriormente, esse trabalho optou por sua utilização no treinamento dos controladores propostos.

3.4 REPOSITÓRIO

Todas as diferentes versões abordadas neste trabalho estão disponíveis no repositório https://github.com/rafaelCabralDS/rl_booster. Em cada caso, encontram-se dois arquivos de configuração: o arquivo env.yaml especifica os hiperparâmetros das recompensas e a definição das condições iniciais do problema, enquanto o arquivo train.yaml refere-se aos hiperparâmetros de treinamento do algoritmo PPO, que é invocado através do arquivo train_sb3.py.

Dentro da pasta booster_env, há arquivos para a definição de constantes, sendo o arquivo env.py responsável pela implementação da API Gymnasium e o ponto de entrada do ambiente implementado. O arquivo world.py utiliza a biblioteca Box2D para implementar os modelos físicos e coordenar as interações da simulação. O arquivo reward.py auxilia nos testes de variações da função de recompensa. Por fim, o arquivo painter.py é responsável pela renderização opcional das simulações.

4 RESULTADOS

São abordados na sequência resultados de algumas variações do mesmo problema, buscando compreender a complexidade de resolução que as condições iniciais e os graus de liberdade tem na capacidade de resolução do algorítimo.

Em média, os primeiros episódios de sucesso começam a ser vistos a partir das 5000 primeiras iterações de treinamento, o que leva em média 3 horas. Na sequência, entre 5000 e 15000 iterações os modelos começam a performar de maneira satisfatória, levando em torno de 10 horas para tal. Após isso, em geral, a sequência do treinamento começa a ser benéfica para o modelo, com as performances decaindo nos testes do agente.

4.1 1 GRAU DE LIBERDADE

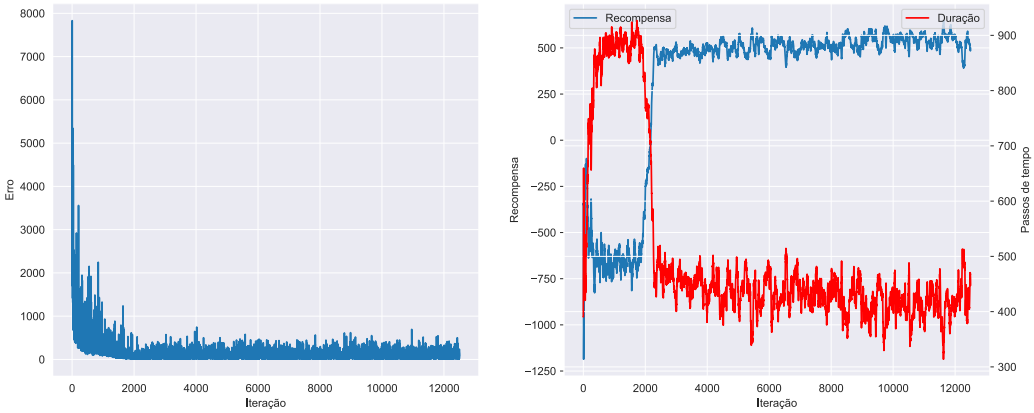
Na tentativa de simular a fase final do pouso, em que o booster está verticalmente alinhado ao launchpad, o controlador é limitado apenas à capacidade de ajustar a potência do motor principal. A condição inicial é formada por altitudes compreendidas entre 600 e 1200 metros, enquanto as velocidades iniciais variam de 80 a 160 m/s. Para o agente, a rede neural é formada por duas camadas, cada uma com 32 nós, uma taxa de aprendizado constante de 0,0005 e um fator de desconto γ de 0,99.

Na Figura 14, à esquerda, é representado o erro total, enquanto à direita são exibidos os valores médios das recompensas e as durações dos episódios extraídos, ambos em função das iterações do algoritmo. Nota-se, através da redução inicial do erro e do pico das recompensas médias, que logo nas primeiras iterações o agente determina a estratégia que seguirá. Após isso, o treinamento passa a otimizar os parâmetros da rede em busca dos comportamentos que maximizam a recompensa total e, por isso, o erro continua oscilando e não converge para zero como se espera em métodos de machine learning comuns.

Em outras palavras, como o agente é o próprio gerador dos dados de treinamento, a medida que sua política atualiza, a forma como interage com o ambiente muda e novas situações são exploradas, resultando em erros maiores que os esperados.

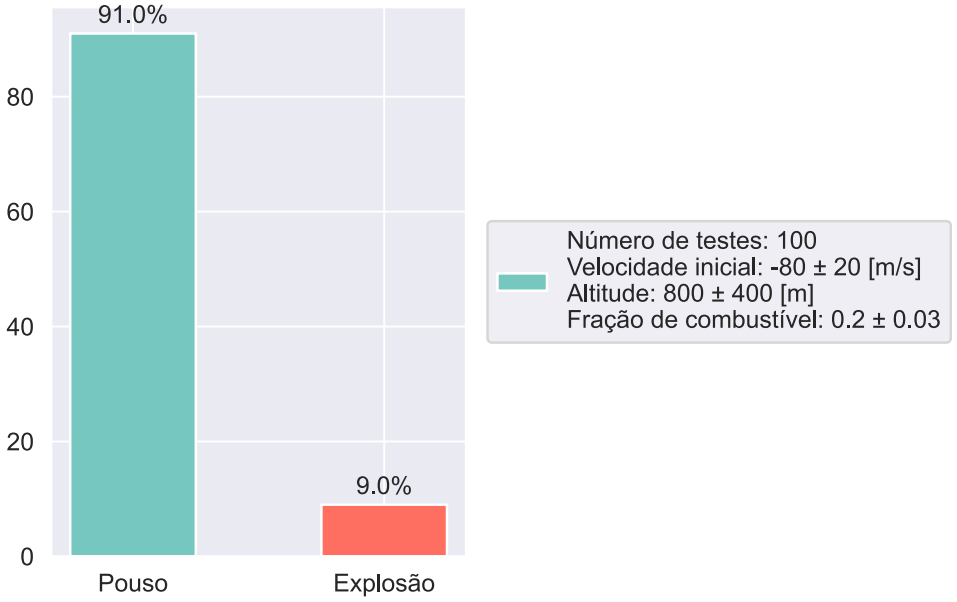
Na figura 15, são mostrados os resultados do desempenho do modelo final obtido, quando submetido a uma amostra de 100 trajetórias em diferentes condições iniciais, compostas por combinações de variações da velocidade, altitude e razão de combustível disponível. Ainda que no caso mais simples, o desempenho obtido não foi de 100%, evidenciando a complexidade do problema a ser resolvido.

Figura 14 – Treinamento do modelo de 1 grau de liberdade



Fonte: Autor

Figura 15 – Desempenho de testes do modelo de 1 grau de liberdade

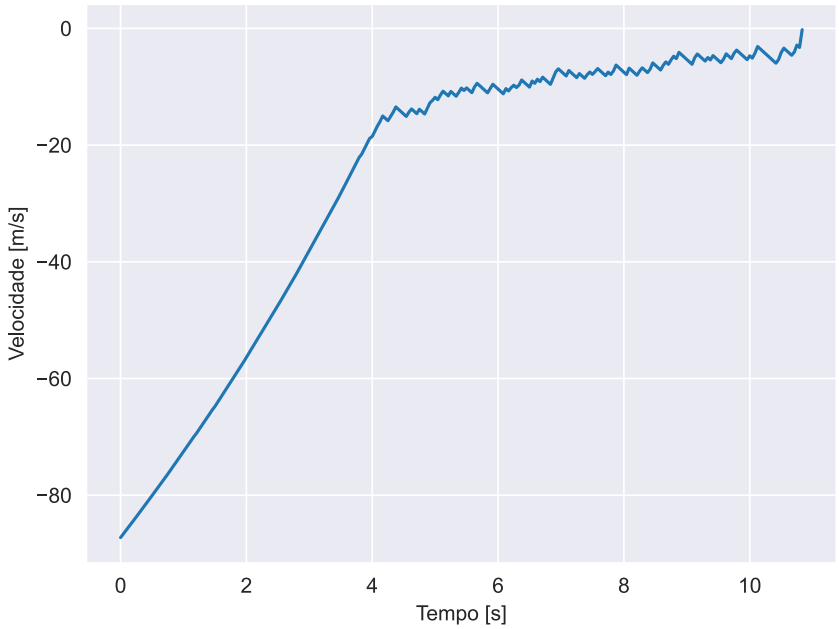


Fonte: Autor

Nas Figuras 16 e 17, respectivamente, é possível observar o perfil de velocidade e as respostas do agente durante o melhor caso na amostra de testes a que o modelo foi submetido. Nota-se que, embora não tenha ocorrido uma única queima — isto é, o agente acionar o motor inicialmente e, posteriormente, apenas regular a potência — foi obtido um perfil contínuo de redução da velocidade.

Finalmente, na figura 18, compara-se o perfil de velocidade de aproximação ao solo da missão SpaceX CRS-11 com o modelo obtido, dadas as mesmas condições

Figura 16 – Perfil de velocidade do melhor caso de teste do modelo de 1 grau de liberdade.



Fonte: Autor

iniciais. Percebe-se que o modelo encontrou uma solução similar ao caso real, apesar das simplificações realizadas. A oscilação observada na velocidade é devido às respostas da potência do agente não serem uniformes.

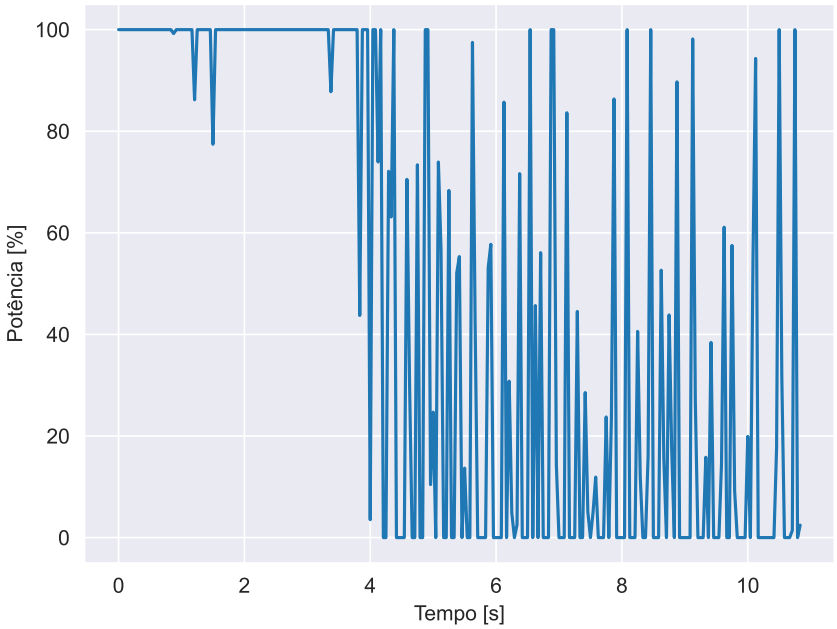
4.2 CASO SEM VETORIZAÇÃO

A fim de diminuir a complexidade do problema, limita-se o controlador à ativação do motor principal, sem a capacidade de vetorização, e dos mecanismos de controle laterais. Dado que a complexidade do problema aumentou, a rede neural é composta por 2 camadas de 64 nós para capturar padrões intrínsecos mais complexos. Além disso, altera-se a estratégia de taxa de aprendizado. Enquanto no caso anterior utilizava-se uma taxa constante durante todo o treinamento, aqui propõe-se uma taxa inicial mais alta de 0,0005 para evitar que o modelo caia em mínimos locais, que decai para 0,0001 durante o treinamento para garantir a estabilidade do aprendizado.

Na Figura 19, observa-se que, ao contrário do modelo anterior, uma rede maior é capaz de reduzir o erro de maneira mais eficiente, atingindo valores próximos de zero rapidamente. Além disso, é nítida a relação inversamente proporcional entre a recompensa e a duração do episódio, devido à existência de um número ótimo de passos a serem tomados na solução ideal.

Na Figura 20, é apresentado o perfil da missão, composto pela trajetória, perfil

Figura 17 – Potência desempenhada no melhor caso de teste do modelo de 1 grau de liberdade.



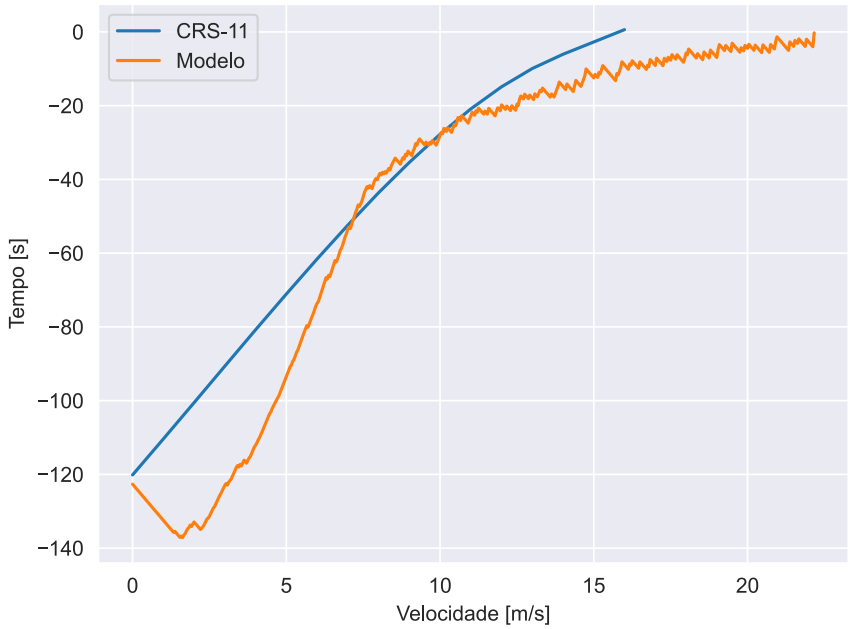
Fonte: Autor

de velocidade e ângulo de pitch, da esquerda para a direita, respectivamente, do melhor episódio observado durante o treinamento do modelo. Observa-se que o padrão de trajetória e velocidade está dentro do esperado, conforme os dados da missão real. No entanto, o comportamento do agente em relação à potência não foi o esperado, uma vez que apresentou um padrão de ligar e desligar o motor repetidamente. A correção desses comportamentos, incentivada pela componente de punições da função de recompensa, é geralmente otimizada após o objetivo principal ser atingido.

Contudo, por razões ainda desconhecidas, os modelos para casos mais complexos, não convergem para a solução encontrada no melhor episódio visto durante o treinamento, mas sim algum outro mínimo local. Compara-se, nas figuras 22 e 23, a missão CRS-11 com uma simulaçao do modelo final obtido, dadas as mesmas condições iniciais.

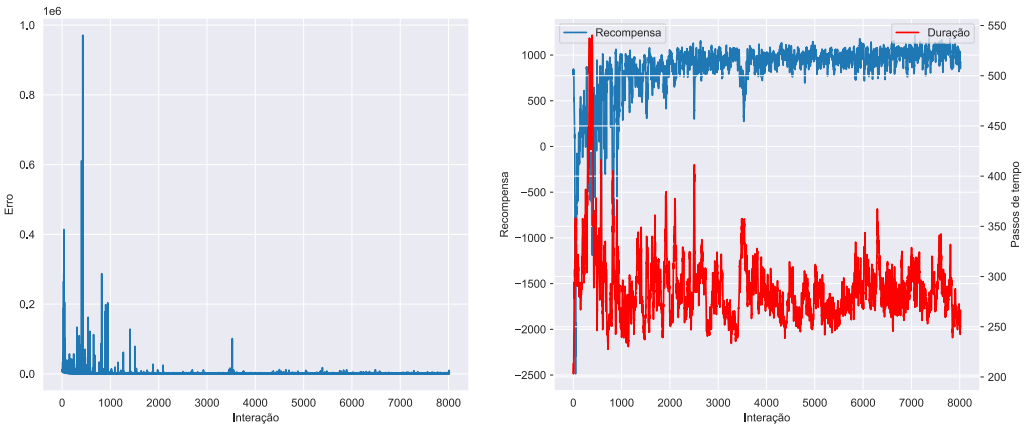
Nota-se que, no caso real, a velocidade horizontal é reduzida mesmo na fase final da aproximação, apesar do ângulo de pitch estar zerado. Isso pode indicar uma inconsistência nos dados da missão ou simplesmente a escolha de utilizar a vetorização para reduzir a velocidade horizontal, em vez de se beneficiar de algum ângulo residual para tal. Na simulação, o agente tenta reduzir a velocidade horizontal tardiamente, resultando em uma maior distância percorrida no eixo x em comparação ao caso real. Este caso foi considerado uma falha, pois o booster toca o solo com uma velocidade vertical de 20 m/s, superior ao limite imposto de 10 m/s.

Figura 18 – Comparação do modelo de 1 grau de liberdade com trecho da missão SpaceX CRS-11



Fonte: Autor

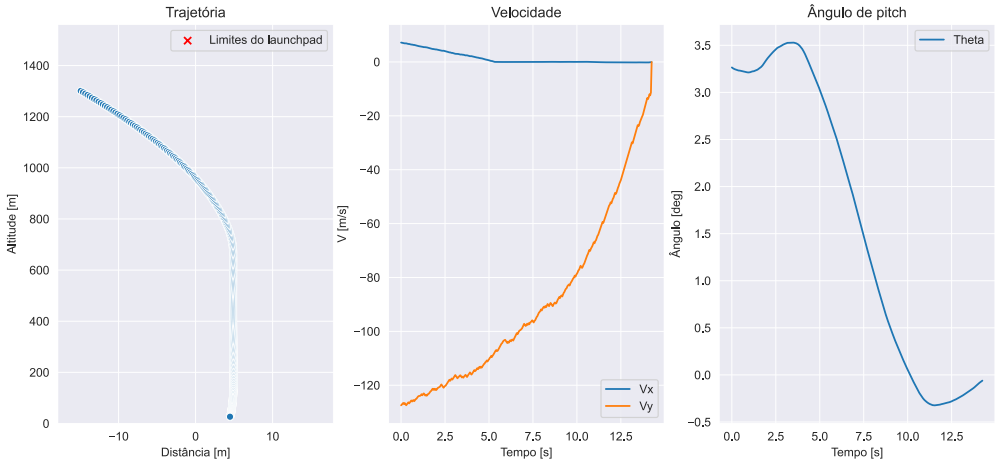
Figura 19 – Dados de treinamento do modelo sem vetorização



Fonte: Autor

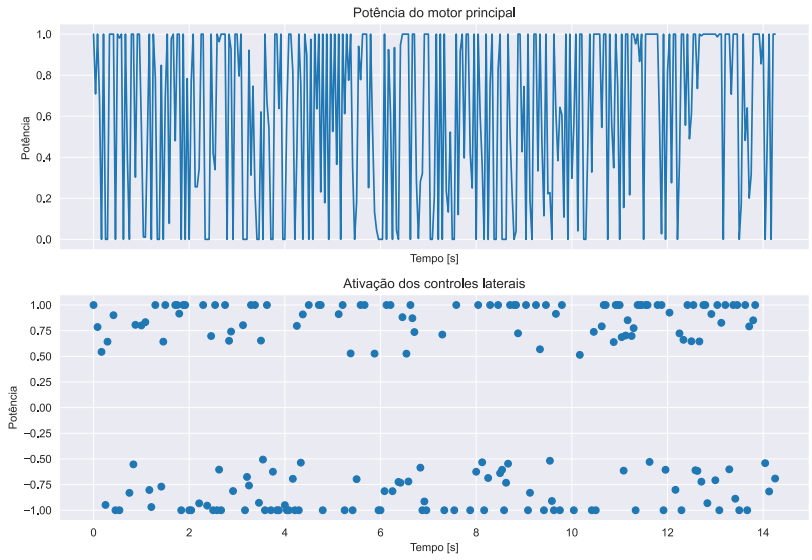
Apresenta-se, na figura 24, o desempenho do modelo sem vetorização, testado para uma amostra de 100 episódios com diferentes combinações iniciais dos parâmetros das componentes da velocidade horizontal e vertical, ângulo de pitch, altura, distância horizontal do alvo e fração de combustível disponível.

Figura 20 – Perfil da missão do melhor episódio visto em treinamento do modelo sem vetorização



Fonte: Autor

Figura 21 – Controle do agente durante o melhor episódio visto em treinamento do modelo sem vetorização

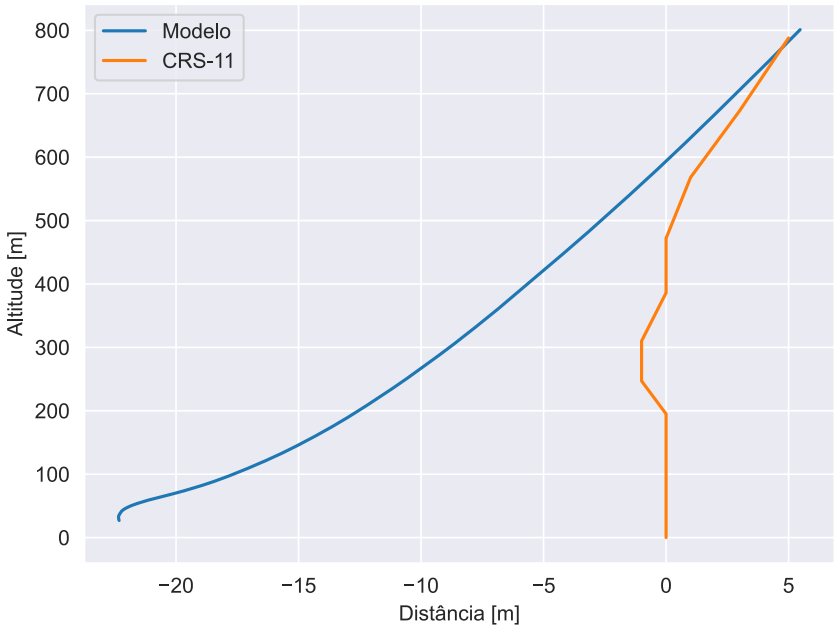


Fonte: Autor

4.3 CASO COMPLETO

Com todas as ações do agente disponíveis, a solução do mesmo problema torna-se mais difícil. Isto porque, ainda que existam mais opções para realizar o controle da aeronave, as redes ganham mais um grau de liberdade na saída. Em consequência, os pesos das redes devem ser alterados de maneira simultânea a fim de otimizar as três saídas de interesse, tornando o problema de aprendizado ainda

Figura 22 – Comparação de trajetória de simulação do modelo sem vetorização com missão SpaceX CRS-11



Fonte: Autor

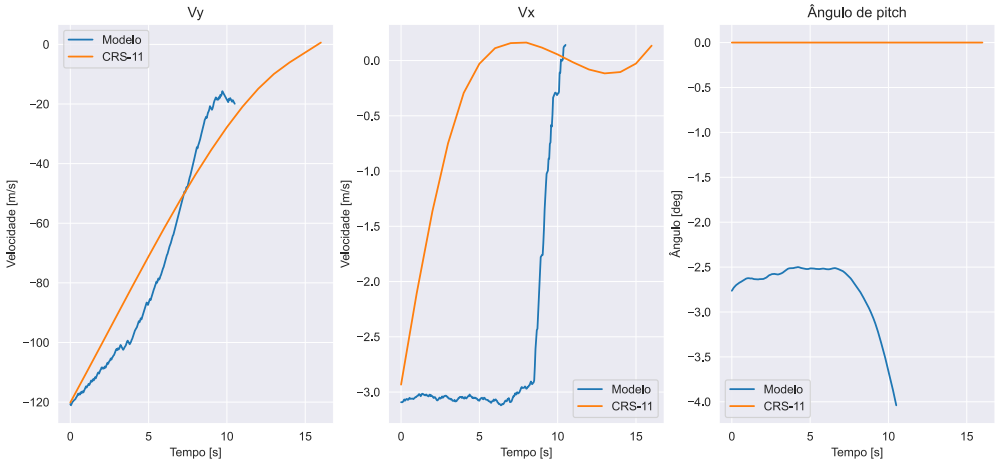
mais complexo.

Algoritmos de otimização de hiperparâmetros podem ser utilizados em casos de aprendizado de difícil solução, onde a busca por combinações ideais de tamanhos de rede, funções de ativação, taxas de aprendizado e outros hiperparâmetros relevantes é realizada de maneira extensiva, visando otimizar algum fator numérico significativo, como a recompensa média do modelo ou a taxa de pousos bem-sucedidos durante os testes dos modelos.

No entanto, essa metodologia é computacionalmente intensiva, uma vez que um único treinamento pode levar mais de 10 horas para começar a apresentar resultados razoáveis. Por essa razão, são utilizadas duas camadas de 64 nós, uma taxa de desconto γ de 0,999 e uma taxa de aprendizado constante de 0,0003, valor padrão recomendado no artigo que apresenta o algoritmo PPO. Na Figura 25, estão apresentadas as curvas de erro, recompensa e duração média dos episódios ao longo das iterações de treinamento.

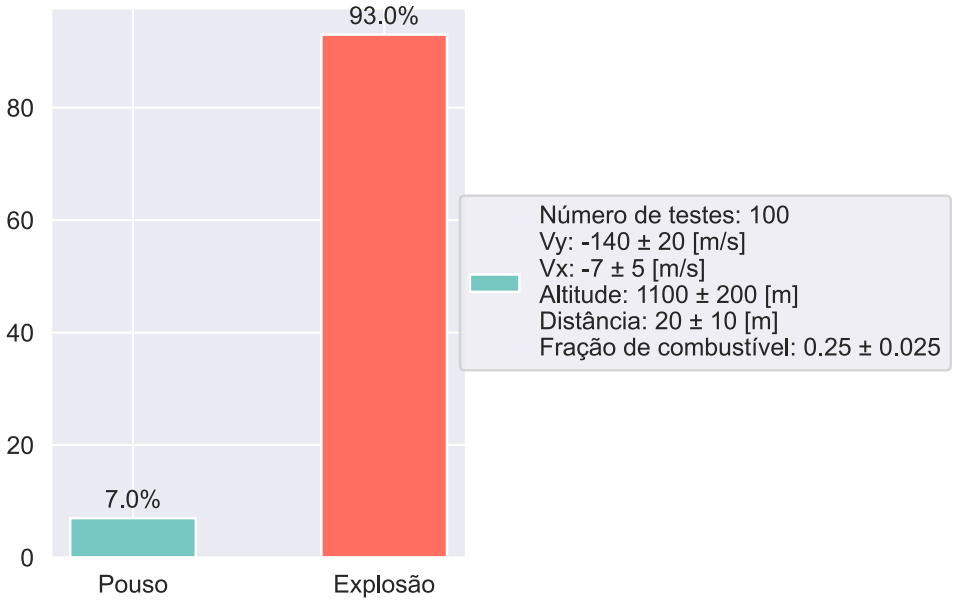
Na Figura 26, é apresentado o perfil da missão, composto pela trajetória, perfil de velocidade e ângulo de pitch, da esquerda para a direita, respectivamente, do melhor episódio observado durante o treinamento do modelo. Observa-se que o agente busca minimizar a velocidade horizontal nos instantes finais, um comportamento que difere da realidade. Contudo, é possível notar uma característica dos algoritmos de aprendizado por reforço, devido à sua natureza exploratória, eles podem encontrar

Figura 23 – Comparação de variáveis de simulação do modelo sem vetorização com missão SpaceX CRS-11



Fonte: Autor

Figura 24 – Performance de testes do modelo sem vetorização

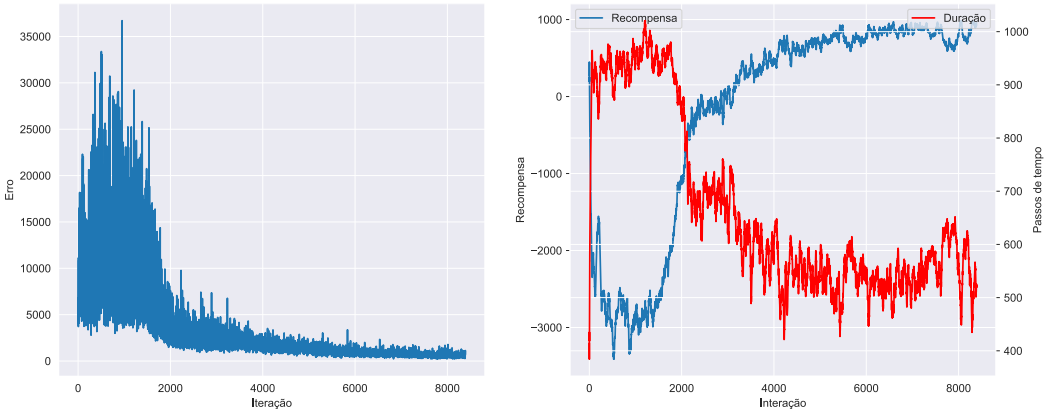


Fonte: Autor

soluções diferentes das esperadas, o que pode ser um ponto positivo para alguns problemas.

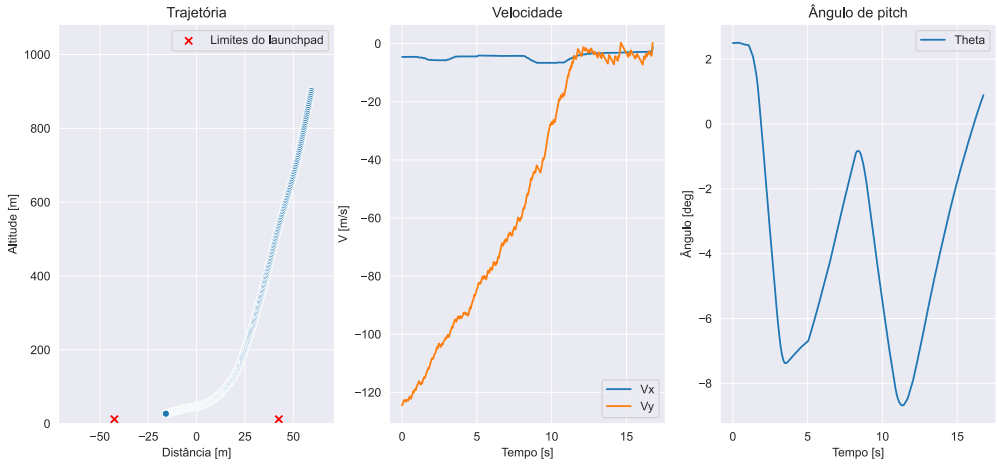
Na Figura 28, compara-se a trajetória, e na Figura 29, comparam-se as velocidades e o ângulo de pitch da missão CRS-11 com uma simulação do modelo final obtido, dadas as mesmas condições iniciais. Apesar das velocidades e do ângulo estarem dentro dos limites aceitáveis para pouso, o agente errou o alvo por apenas

Figura 25 – Dados de treinamento do modelo completo



Fonte: Autor

Figura 26 – Perfil da missão do melhor episódio visto em treinamento do modelo completo



Fonte: Autor

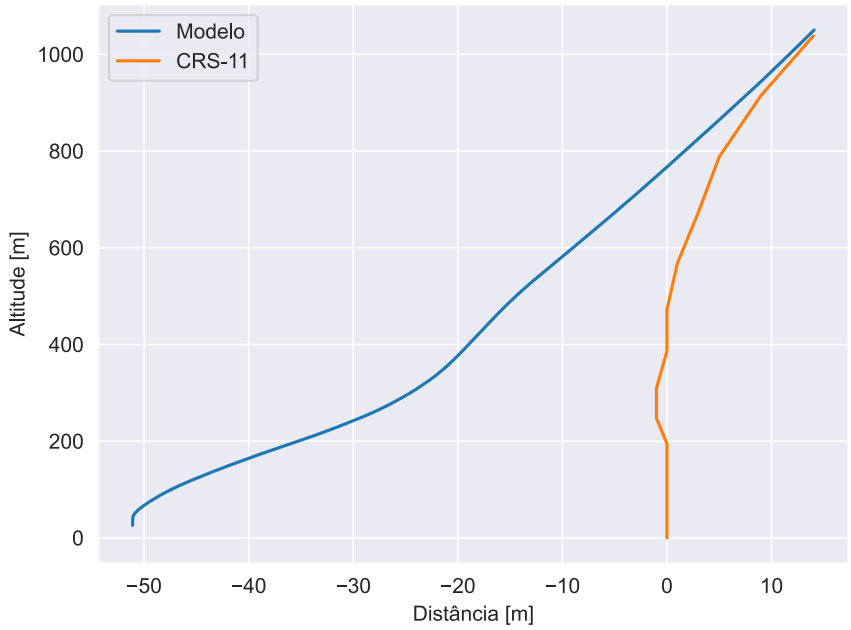
alguns metros, resultando em uma explosão. Observa-se que a velocidade vertical foi reduzida de maneira similar ao caso real. Além disso, o agente utilizou o ângulo de pitch para controlar a velocidade horizontal.

Apresenta-se, na figura 30, o desempenho do modelo final testado para uma amostra de 100 episódios com diferentes combinações iniciais dos parâmetros das componentes da velocidade horizontal e vertical, ângulo de pitch, altura, distância horizontal do alvo e fração de combustível disponível.

Figura 27 – Controle do agente durante o melhor episódio visto em treinamento do modelo completo

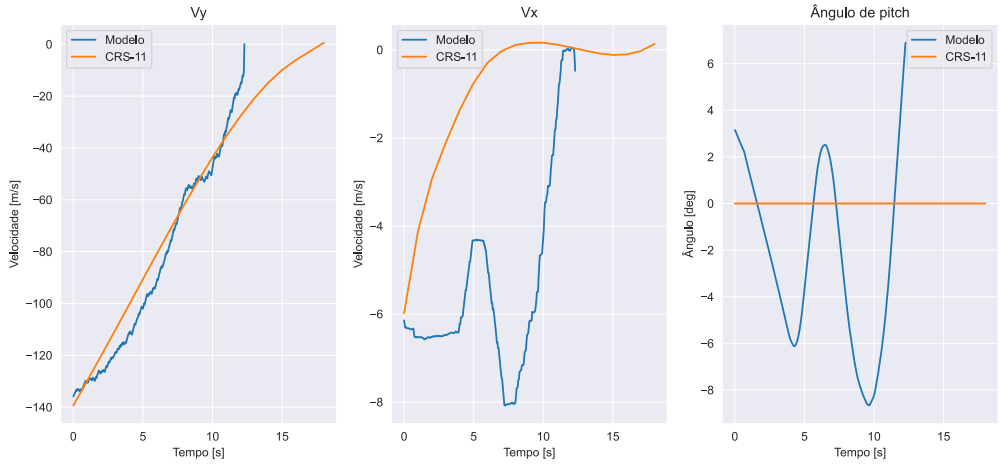
Fonte: Autor

Figura 28 – Comparação da trajetória de simulação do modelo completo com a missão SpaceX CRS-11



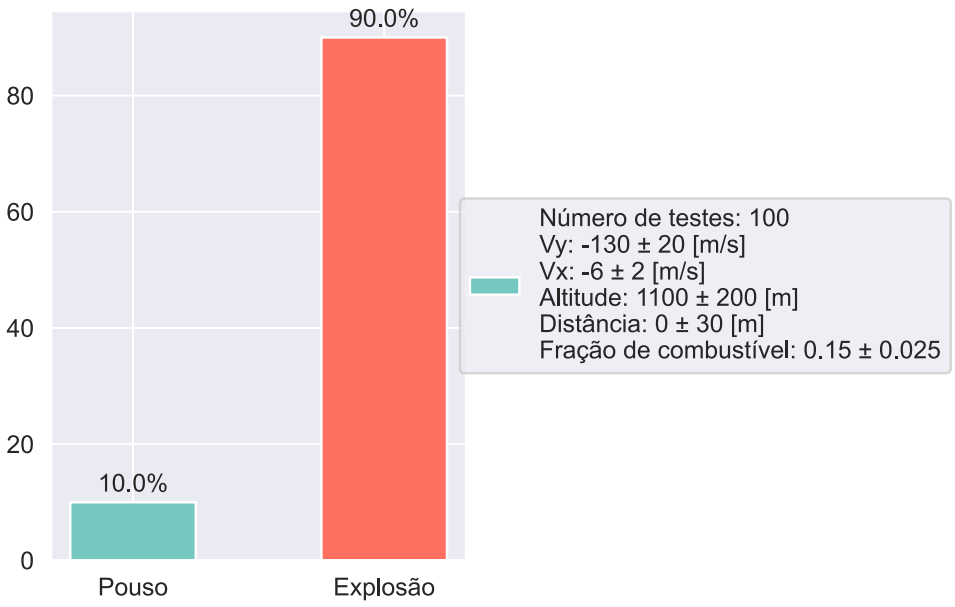
Fonte: Autor

Figura 29 – Comparação de variáveis de simulação do modelo completo com a missão SpaceX CRS-11



Fonte: Autor

Figura 30 – Performance de testes do modelo completo



Fonte: Autor

5 CONCLUSÃO

Durante os treinamentos, apesar de algumas soluções promissoras terem sido identificadas, os modelos convergiram para soluções subótimas. Isso pode ser atribuído a vários fatores, incluindo a alta dimensionalidade do espaço de estados e ações e as modelagens de recompensas testadas. Essas condições dificultam a estabilidade do aprendizado e a consistência na convergência para políticas ótimas, resultando em desempenhos não satisfatórios.

A grande demanda por poder de processamento para treinar os modelos resultou em tempos de treinamento prolongados. Além disso, a eficiência dos algoritmos de RL é altamente sensível a hiperparâmetros de treinamento, no qual o ajuste buscando a convergência dos modelos nem sempre é trivial e podem não ser otimizados de forma eficaz devido a essas limitações computacionais.

As limitações na modelagem do ambiente de simulação também desempenharam um papel significativo nas dificuldades encontradas. Ambientes de simulação muitas vezes não conseguem capturar todas as complexidades e variabilidades do mundo real, levando a uma variancia entre o desempenho simulado e o desempenho no mundo real. Além disso, a fidelidade da simulação pode ser comprometida por simplificações necessárias para viabilizar a implementação do ambiente, o que pode levar a um modelo que não generaliza bem quando aplicado a situações fora das simulações.

Por fim, embora o estudo não tenha conseguido utilizar RL para desenvolver controladores eficientes para realizar pousos de foguetes, a pesquisa demonstra que essa abordagem é viável. A aplicação de RL em sistemas de controle de foguetes possui um enorme potencial, conforme mostrado nos resultados dos melhores casos vistos durante o treinamento, que foram capazes de reproduzir trajetórias e padrões similares com as vistas na missão CRS-11. Apesar das dificuldades encontradas neste estudo, avanços contínuos em técnicas de RL e melhorias na modelagem de simulação podem eventualmente permitir que RL seja efetivamente utilizado para esse fim, contribuindo significativamente para a área de controle de veículos espaciais.

REFERÊNCIAS

ANDERSON, J. D. J. **Hypersonic and High-Temperature Gas Dynamics**. 2nd. ed. [S.l.]: American Institute of Aeronautics and Astronautics, Inc., 2006.

BARTO, A.; ROSS, R. **Reinforcement Learning: An Introduction**. 2. ed. [S.l.]: MIT press, 2018.

BELLMAN, R. **DYNAMIC PROGRAMMING**. 7. ed. [S.l.]: PRINCETON UNIVERSITY PRESS, 1957.

BRUNTON, S. L.; KUTZ, J. N. Data-driven science and engineering: Machine learning, dynamical systems, and control. Cambridge University Press, 2019.

CATTO, E. **Box2D Documentation**. [S.l.], 2023. A 2D physics engine for games. Disponível em: https://box2d.org/documentation/.

DREIER, C. An improved cost analysis of the apollo program. **Space Policy**, v. 60, p. 101476, 2022. ISSN 0265-9646. Disponível em: https://www.sciencedirect.com/science/article/pii/S0265964622000029.

FARAMA. **Lunar Lander**. 2024. Accessed: 2024-05-15. Disponível em: https://gymnasium.farama.org/environments/box2d/lunar_lander/.

FERRANTE, R. **A Robust Control Approach for Rocket Landing**. 2017. Dissertação (Master of Science Artificial Intelligence) — University of Edinburgh, 2017.

FRANKLIN, G. F.; POWELL, J. D.; EMAMI-NAEINI, A. **Feedback Control of Dynamic Systems**. 6th. ed. [S.l.]: Pearson, 2019.

FUKUSHIMA, K. A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. **Biological Cybernetics**, v. 36, 1980.

GARDI, J.; ROSS, J. **An Illustrated Guide to SpaceX's Launch Vehicle Reusability Plans**. 2019. http://www.justatinker.com/Future/. Accessed: 2023-05-29.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural Networks**, v. 2, n. 5, 1989.

KHATAL, A. et al. Rocket science for the space race. *In*: INTERNATIONAL CONFERENCE ON COMMUNICATION AND INFORMATION PROCESSING (ICCIP-2023). [S.l.], 2023.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, n. 7553, 2015.

MOOS, J. et al. Robust reinforcement learning: A review of foundations and recent advances. MDPI, 2022. Disponível em: https://doi.org/10.3390/make4010013.

O'CONNOR, A. C. et al. **Economic Benefits of the Global Positioning System (GPS) Final Report**. [S.l.], 2019. RTI Project Number 0215471.

OPENAI. **INTRODUCTION TO RL**. 2018. https://spinningup.openai.com/en/latest/spinningup/rl_intro.html. Accessed: 2023-06-20.

OPENAI. **Proximal Policy Optimization**. 2018. Disponível em: https://spinningup.openai.com/en/latest/algorithms/ppo.html.

PLAPPERT, M. et al. Parameter space noise for exploration. 2017.

STUART J. RUSSELL DAISHI HARADA, A. N. Policy invariance under reward transformations: Theory and application to reward shaping. *In*: **International Conference on Machine Learning**. [s.n.], 1999. Disponível em: https://api.semanticscholar.org/CorpusID:5730166.

SCHULMAN, J.; WOLSKI, F.; DHARIWAL, P.; RADFORD, A.; KLIMOV, O. Proximal policy optimization algorithms. 2017. OpenAI. Disponível em: https://arxiv.org/abs/1707.06347.

SHALEV, S. **Telemetry Data**. 2024. Disponível em: https://github.com/shahar603/Telemetry-Data.

SPACEX. **Falcon User's Guide**. [S.l.], 2021. Disponível em: https://www.spacex.com/media/falcon-users-guide-2021-09.pdf.

TOMANEK, R.; HOSPODKA, J. Reusable launch space systems. **Magazine of Aviation Development**, 2018.

WALL, M. **SpaceX Lands Orbital Rocket Successfully in Historic First**. 2015. https://www.space.com/31420-spacex-rocket-landing-success.html. Accessed: 2023-06-13.