UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

JEYDSON LOPES DA SILVA

DESENVOLVIMENTO DE CONTROLADOR BASEADO EM APRENDIZADO EMOCIONAL PROFUNDO

Recife
2021

JEYDSON LOPES DA SILVA

DESENVOLVIMENTO DE CONTROLADOR BASEADO EM APRENDIZADO EMOCIONAL PROFUNDO

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, como requisito parcial para obtenção do título de doutor em Engenharia Elétrica. Área de concentração: Processamento de Energia.

Orientador: Prof. Dr. Ronaldo Ribeiro Barbosa de Aquino

Coorientadora: Profa. Dra. Aida Araújo Ferreira

Recife

2021

Catalogação na fonte
Bibliotecária Sandra Maria Neri Santiago, CRB-4 / 1267

S586d Silva, Jeydson Lopes da.
Desenvolvimento de controlador baseado em aprendizado emocional profundo / Jeydson Lopes da Silva. – Recife, 2021.
210 folhas, il., figs., tabs.

Orientador: Prof. Dr. Ronaldo Ribeiro Barbosa de Aquino.
Coorientadora: Profa. Dra. Aida Araújo Ferreira.
Tese (Doutorado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia Elétrica, 2021.
Inclui Referências e Apêndices.

1. Engenharia Elétrica. 2. Aprendizado por reforço. 3. Aprendizado profundo. 4. Aprendizado emocional. 5. Controlador emocional. I. Aquino, Ronaldo Ribeiro Barbosa de (Orientador). II. Ferreira, Aida Araújo (Coorientadora). III. Título.

UFPE

621.3 CDD (22. ed.) BCTG/2021-161

JEYDSON LOPES DA SILVA

DESENVOLVIMENTO DE CONTROLADOR BASEADO EM APRENDIZADO EMOCIONAL PROFUNDO

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, como requisito parcial para obtenção do título de doutor em Engenharia Elétrica. Área de concentração: Processamento de Energia.

Aprovado em: 12/04/2021.

BANCA EXAMINADORA

_____
Profº. Dr. Ronaldo Ribeiro Barbosa de Aquino (Orientador)
Universidade Federal de Pernambuco

_____
Profº. Dr. Francisco de Assis dos Santos Neves (Examinador Interno)
Universidade Federal de Pernambuco

_____
Profª. Dra. Aida Araújo Ferreira (Examinador Externo)
Instituto Federal de Pernambuco

_____
Profº. Dr. Manoel Afonso de Carvalho Júnior (Examinador Externo)
Universidade Federal de Pernambuco

_____
Profº. Dr. Cleber Zanchettin (Examinador Externo)
Universidade Federal de Pernambuco

_____
Profª. Dra. Milde Maria da Silva Lira (Examinador Externo)
Universidade Federal de Pernambuco

Dedico aos meus pais, esposa e avós Edna França e Jaime Joaquim *(in memoriam)*.

AGRADECIMENTOS

Primeiramente agradeço a Deus, o qual proporcionou todos os meios e oportunidades para a realização dos meus objetivos em vida. A minha amada esposa Julyane que sempre dedicou seu apoio e amor, tendo uma fé inabalável em Deus e muita confiança em mim, proporcionando-me todas as forças para a realização do trabalho. A minha amada mãe Maria, a qual em toda sua vida dedicou-me um grande amor incondicional, fornecendo desde o meu nascimento a educação e o caráter que me guiam até hoje. Ao meu pai Jaime e meu irmão Jefferson por todo o companheirismo e ensinamentos ao longo da vida. A família Freitas, a qual me abraçou ainda na juventude e tornou possível o meu desenvolvimento em todas as áreas de minha vida. A família França, a qual foi sempre uma representação de amor na minha vida. Ao meu orientador Prof. Ronaldo pela fé depositada em mim e no meu trabalho e, principalmente, os conselhos, orientações e ensinamentos para toda a vida. Aos professores do LDSP e LEEQE pelos constantes apoios e ensinamentos vivenciados ao longo do tempo. Ao meu companheiro de laboratório Davidson, o qual foi de suma importância para a realização deste e outros trabalhos. Aos professores e colegas do departamento de engenharia elétrica da UFPE, os quais me inspiram, incentivam e ensinam a me tornar um melhor profissional.

RESUMO

Os controladores biologicamente inspirados demonstram grande êxito em diversas aplicações, principalmente em situações que apresentam perturbações e incertezas nas dinâmicas do sistema. Nos últimos tempos, surgiram diversos trabalhos concernentes à área do aprendizado do cérebro humano, permitindo assim o surgimento de novas teorias e aplicações na engenharia de controle. Nesse âmbito, controladores baseados no aprendizado emocional que ocorre no cérebro humano são capazes de oferecer novos recursos e resultados satisfatórios em termos da dinâmica de resposta de controle. Este tipo de controlador é associado a capacidade de reação dos seres humanos aos estímulos sensoriais externos e ao consequente dilema entre a decisão baseada na emoção ou razão. Todavia, a concepção e o comissionamento deste tipo de controlador ainda representam um grande desafio para os pesquisadores, uma vez que é necessário a determinação de alguns sinais característicos a este sistema (estímulos), os quais podem variar de aplicação para aplicação. Portanto, a utilização de novas ferramentas matemáticas e computacionais podem oferecer os recursos necessários para a melhora de desempenho deste seguimento de controlador. Nesse sentido, o presente trabalho propõe a utilização de algoritmos baseados em aprendizado por reforço, associados aos recentes avanços na área do aprendizado profundo das redes neurais, tendo por objetivo a elaboração de novas arquiteturas que permitam a este controlador atingir uma maior generalização em sua aplicação, bem como fornecer uma alternativa viável aos modelos tradicionais em uso. Além disso, é proposta uma metodologia para o desenvolvimento, comissionamento e uso final desta nova proposta de controlador emocional, destacando-se as ferramentas e recursos necessários a esta tarefa. Por fim, a partir de diversos resultados de simulações experimentais, bem como aplicação real, busca-se demonstrar a eficácia da proposta da união do controlador emocional e os recursos advindos da teoria do aprendizado por reforço profundo.

Palavras-chave: aprendizado por reforço; aprendizado profundo; aprendizado emocional; controlador emocional.

ABSTRACT

Biologically inspired controllers demonstrate great success in several applications, mainly in situations that present disturbances and uncertainties in the system dynamics. In recent times, several works have appeared in the area of learning the human brain thus allowing the emergence of new theories and applications in control engineering. In this context, controllers based on the emotional learning that takes place in the human brain are able to offer new resources and satisfactory results in terms of the control response dynamics. This type of controller is associated with the ability of human beings to react to external sensory stimuli and the consequent dilemma between the decision based on emotion or reason. However, the design and commissioning of this type of controller still represents a great challenge for researchers, since it is necessary to determine some characteristic signals to this system (stimuli), which can vary from application to application. Therefore, new mathematical and computational tools can offer the necessary resources to improve the performance of this controller segment. In this case, present work proposes the use of reinforcement learning-based algorithms, associated with recent advances in the area of deep learning of neural networks, aiming at the development of new architectures that allow this controller to achieve greater generalization in its application, as well as providing a viable alternative to traditional models in use. In addition, a methodology is proposed for the development, commissioning and end use of this new proposal for an emotional controller, highlighting the tools and resources needed for this task. Finally, from various results of experimental simulations, as well as real application, we seek to demonstrate the effectiveness of the proposal of the union of the emotional controller and the resources derived from the theory of learning by deep reinforcement.

Keywords: reinforcement learning; deep learning; emotional learning; emotional controller.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A2C	Advantage Actor-Critic
A3C	Asynchronous Advantage Actor-Critic
AC	Actor-Critic
ACER	Sample Efficient Actor-Critic with Experience Replay
ACKTR	Actor-Critic using Kronecker-Factored Trust Region
AI	Artificial Intelligence
ANNs	Artificial Neural Networks
ARM	Advanced RISC Machine
BEL	Brain Emotional Learning
BELBIC	Brain Emotional Learning Based Intelligent Controller
DBELBIC	Deep Brain Emotional Learning Based Intelligent Controller
DDPG	Deep Deterministic Policy Gradient
CC	Corrente Contínua
CLP	Controlador Lógico Programável
DA	Data Acess
DB	Data Block
DDPGs	Deep Deterministic Policy Gradients
DL	Deep Learning
DNN	Deep Neural Networks
DP	Dynamic Programming
DRL	Deep Reinforcement Learning
DQN	Deep Q-learning Network
EL	Emotional learning
ELETROBRÁS	Centrais Elétricas Brasileiras S.A

FLC	Fuzzy Logic Controller
GA	Genetic Algorithm
G-BELBIC	Generalized Brain Emotional Learning Based Intelligent Controller
HJB	Hamilton-Jacobi-Bellman
IPMSM	Interior Permanent Magnet Synchronous Motor
LSTM	Long short-term memory
LAMOTRIZ	Laboratório de Sistemas Motrizes
LEEQE	Laboratório de Eficiência Energética e Qualidade de Energia
LR	Lugar das Raízes
MC	Monte Carlo
MDP	Markov Decision Processes
MIPS	Microprocessor without interlocked pipeline
ML	Machine Learning
MLP	Multilayer perceptron
MO	Model Output
NGD	Natural Gradiente Descendente
NLP	Natural language processing
OC	Orbitofrontal cortex
OPC	Object Linking and Embedding for Process Control
PI	Proportional-Integer
PID	Proportional-Integer-Derivative
PPO	Proximal Policy Optimization
PROCEL	Programa Nacional de Conservação de Energia Elétrica
PSO	Particle Swarm Optimization
RL	Reinforcement Learning
R	Reward

S	Sensorial Input
SAC	Soft Actor-Critic
SISO	Single Input Single Output
SoC	System-on-a-chip
TF	Tensorflow
TD	Temporal Differences
TD3	Twin Delayed DDPG
TRPO	Temporal Region Policy Optimization

LISTA DE SÍMBOLOS

$\doteq$	Relação de igualdade que é verdadeira por definição
s', s	Estados
a	Ação
r	Recompensa
$\mathcal{S}$	Espaço de estado
$\mathcal{A}(s)$	Subespaço de ação no estado s
$\mathcal{A}$	Espaço de ação
$\mathcal{R}$	Espaço de recompensa
t	Época de decisão ou instante de tempo discreto
T	Passo de tempo final de um episódio
S_t	Estado no instante de tempo t
A_t	Ação no instante de tempo t
R_t	Recompensa no instante de tempo t
π	Política
δ	Erro no algoritmo TD
$h(a\|s)$	probabilidade de se tomar uma ação a no estado s sob uma política π estocástica.
$p(s', r\|s, a)$	probabilidade de transição para o estado s' com recompensa r, do estado s tomando uma ação a estocástica.
$p(s'\|s, a)$	probabilidade de transição para o estado s', do estado
$r(s, a)$	Recompensas esperadas para um par estado-ação
$r(s, a, s')$	Recompensas esperadas para a tripla relação estado-ação-próximo estado s tomando uma ação a.
E_π	Valor esperado de uma variável a partir de uma política π
G_t	Retorno (descontado) acumulado a partir do instante de tempo t

γ	Fator de desconto
$\upsilon_\pi(s)$	Valor do estado s sob uma política π
$\upsilon^*(s)$	Valor do estado s sob uma política ótima.
$q_\pi(s,a)$	Valor de se tomar uma ação a no estado s sob uma política π
$q^*(s,a)$	Valor de se tomar uma ação a no estado s sob uma política ótima
J	Função de recompensa
d_π	Distribuição estacionária da cadeia de Markov
$\pi_\theta(a\|s)$	Política modelada a partir de uma função parametrizada em relação θ;
$\mathcal{D}$	*Experience Replay*
$\mathcal{N}$	Fator de exploração de *Ornstein-Unlenbeck*
ω	Parâmetros de *Q* no AC
L	Função *Loss*
τ	Parâmetro relacionado a uma atualização suave na política no DDPG
ε	Limitação de ruídos da ação no TD3
B	*minibatch*
$Q(s,a)$	Função valor estado-ação
$\mu(s)$	Ação determinística
ρ	Distribuição de estado com desconto
μ_θ	Política determinística
σ	Variável de variação estocástica
α	Taxa de aprendizagem da amígdala no BELBIC
α_Q	Taxa de aprendizagem no *Q-learning*
β	Taxa de aprendizagem do córtex no BELBIC
ΔV	Ganho da amígdala
ΔW	Ganho do córtex orbitofrontal
θ	Parâmetros da política do agente de RL

A_i	Sinal da amígdala
A_{th}	Sinal máximo da amígdala
MO	Saída do modelo
O_i	Sinal do córtex
S	Sinal do estímulo sensorial
R	Sinal do estímulo emocional
V	Peso da amígdala
W	Peso do córtex
K	Ganhos nos estímulos do DBELBIC

SUMÁRIO

1 INTRODUÇÃO

A maioria dos sistemas de controle no ramo da engenharia, bem como diversos aspectos comportamentais dos seres vivos, utilizam-se de processos baseados em controles com realimentação contínua para a tomada de decisão em tempo real (OGATA, 2003; DORF; BISHOP, 2018). São vários os exemplos de sistemas que se utilizam de realimentação para executar ações. Nos seres humanos, a manutenção adequada da temperatura corporal, por exemplo, baseia-se em um controle robusto com realimentação. Esse sistema permite que os limites de temperatura sejam mantidos, apesar da grande variabilidade e distúrbios de temperaturas do ambiente externo. Tal regulação ocorre com o monitoramento da temperatura e as devidas ações de controle, como o aumento do metabolismo e sudorese. No tocante aos pensamentos e ações, ambos estão intimamente relacionados em uma arquitetura de realimentação bastante densa nos sistemas nervoso e cerebral. Os estímulos externos são coletados, assimilados e, a partir disso, são tomadas decisões, consequentemente, ações de controle são executadas, permitindo a interação das pessoas com o mundo.

Uma das principais e necessárias características nos projetos dos sistemas de controle é a estabilidade. Existem diversas formas de abordar tal situação nos projetos de controladores que, muitas vezes, atuam em situações que envolvem dinâmicas não lineares do processo o qual se pretenda controlar. Essa situação acaba por dificultar a obtenção de soluções ideais, visto que a maioria das técnicas de resolução dos problemas, relativamente à teoria de controle, baseiam-se em otimização de problemas lineares. Para os sistemas, sejam lineares ou não lineares, a estabilidade é um aspecto essencial no que diz respeito a teoria da otimalidade do controlador (WANG et al., 2008). De forma geral, o problema da otimalidade do controlador consiste em projetar um controlador que minimiza uma medida do comportamento de um sistema dinâmico no tempo (SUTTON; BARTO, 2018).

No que diz respeito ao campo de pesquisa do controle ótimo, considera-se como o marco inicial, os estudos desenvolvidos por Bellman[1] e Pontryagin[2] (BUSONIU et al., 2010), datados do final da década de 1930. Esta área permite diversas possibilidades para o desenvolvimento de controladores, as quais podem variar desde, sistemas lineares com funções de custos quadráticas até sistemas que apresentam grandes características de não linearidades, que por sua vez necessitam de controladores mais complexos. Em relação a um sistema linear quadrático, a solução completa é possível de obter-se quando da utilização da equação de Riccati[3]. Em contrapartida, no caso de um sistema não linear, existe uma classe de métodos denominada de programação dinâmica - dynamic programming (DP) (BELLMAN, 1957; BELLMAN, 1958; HOWARD, 1960), a qual provê as condições matemáticas necessárias para a obtenção de um

[1] Richard Ernest Bellman foi um matemático norte-americano (1920 - 1984).
[2] Lev Semenovich Pontryagin foi um matemático russo (1908 - 1988).
[3] Jacopo Francesco Riccati foi um matemático e físico italiano (1676 - 1754).

controlador ótimo (LENDARIS, 2009). Bellman demonstrou que em um problema de controle ótimo, uma estratégia que pode ser abordada é a determinação da solução de uma “equação funcional básica”, conhecida como a equação de otimalidade de Bellman, ou equação *Hamilton-Jacobi-Bellman* (HJB). A partir disso, Howard[4] propôs um importante teorema para a melhoria da política de decisões nos algoritmos de aprendizado por reforço - *reinforcement learning* (RL). Além disso, foi responsável pelo desenvolvimento de um algoritmo de iteração de política para processos de decisão markovianos - *markov decision processes* (MDP) (HOWARD, 1960).

Nos casos em que existam modelos de sistemas disponíveis, a DP pode ser considerada uma forma bastante eficaz na resolução de problemas que envolvam um controle ótimo (SUTTON; BARTO, 2018; BERTSEKAS, 1987; BUSONIU et al., 2010). No entanto, deve-se considerar a "maldição da dimensionalidade" (*curse of dimensionality*), ou seja, a quantidade de elementos necessários para um treinamento adequado de um classificador é, em muitos casos, uma função exponencial, $O(e^N)$ (JAIN; DUIN; MAO, 2000). Apesar desta situação, a DP é considerada uma ferramenta do estado da arte na resolução dos problemas de controle ótimo, caso haja a disponibilidade do modelo do sistema (BUSONIU et al., 2010). De forma geral, a DP lida com o problema de como encontrar controladores ótimos em sistemas dinâmicos sujeitos a algum grau de aleatoriedade.

No que diz respeito as situações onde não haja a disponibilidade de um modelo do sistema, soluções alternativas foram propostas na década de 1960. (MENDEL, 1966; MINSKY, 1963; WALTZ; FU, 1965). Todavia, este campo de estudos apenas floresceu cerca de 20 anos depois, quando do desenvolvimento dos estudos na área do RL (BARTO; SUTTON; ANDERSON, 1983; SUTTON, 1984; SUTTON, 1988; WATKINS, 1989; WERBOS, 1987). Os estudos que contemplam a área do aprendizado por reforço caracterizam-se pelo desenvolvimento de algoritmos que, principalmente, aprendam políticas de controle por meio de amostras de transição entre os estados do sistema, as quais podem ser coletadas antecipadamente (*offline*) ou pela interação em tempo real com o sistema (*online*) (BARRETO et al., 2020). Apesar do termo aprendizado por reforço ser tratado em diferentes contextos, dependendo da área de estudo, pesquisas com o RL moderno denotam de uma síntese de décadas atrás, unindo as ideias de controle ótimo, aprendizado animal e métodos de inteligência artificial (WILLIAMS, 2009a).

Na área da psicologia, as teorias de reforço, concernentes às etapas de aprendizado animal, são muito importantes para formação das bases desse ramo da ciência. Sabe-se há tempos que a atribuição de certas recompensas ou punições por determinadas ações possibilita a modificação do comportamento de um animal. Baseado nisso, (THORNDIKE, 1898) propôs os primeiros estudos com a teoria de aprendizado baseada na "tentativa e erro", ou seja, caso uma resposta de comportamento tenha um resultado favorável, as conexões neurais que proporcionaram tal situação são reforçadas. Nesse contexto, (PAVLOV, 1927) foi quem primeiramente utilizou o termo "reforço" em seu trabalho, o qual utilizou estímulos de recompensas ou punições para

[4] Ronald Arthur Howard é professor na escola de engenharia da universidade de Stanford.

alterar padrões comportamentais de cachorros por meio da indução condicional de reflexos.

Nos início do desenvolvimento das teorias da inteligência artificial - *artificial intelligence* (AI), credita-se o surgimento das primeiras ideias de associar o aprendizado por meio de tentativa e erro com a computação. Alan Turing[5] descreveu em seus trabalhos iniciais na área de inteligência computacional um projeto para um "sistema de prazer e dor", o qual funcionava de acordo com a reação aos estímulos (TURING, 1950). No entanto, foi apenas nos anos 1960 que se abordou o RL na área de controle e engenharia (WALTZ; FU, 1965; MENDEL; MACLAREN, 1970; MENDEL, 1966; FU, 1970). Tal fato pode estar relacionado, em grande parte, ao famoso trabalho publicado por Minsky[6], "*Steps toward artificial intelligence*", trazendo à tona várias situações de relevância para a área do RL (MINSKY, 1963).

Uma das principais dificuldades encontradas na área de estudo do RL, relaciona-se com a complexidade na representação das soluções exatas dos sistemas que possuam espaços de estados e ação contínuos, ou discretos com uma alta dimensionalidade, como no caso da maioria dos problemas que envolvem sistemas de controles dinâmicos (BUSONIU et al., 2010; PRECUP; SUTTON; DASGUPTA, 2001). Neste caso, uma solução possível se baseia na representação compacta de tais soluções, por exemplo, pode-se fazer uso dos aproximadores de funções. Em relação a essa questão, os crescentes avanços na área do RL têm permitido, cada vez mais, que os métodos utilizados aproximem-se das dimensões dos problemas reais (DULAC-ARNOLD; MANKOWITZ; HESTER, 2019).

No contexto dos aproximadores de função, aqueles que são mais amplamente considerados, referencialmente aos algoritmos de RL, são os mapeadores lineares e as redes neurais artificiais - *artificial neural networks* (ANNs) (WILLIAMS, 2009b). A utilização das ANNs na aproximação de funções representa uma poderosa ferramenta, o que justifica o fato de serem amplamente utilizadas para aproximações de funções não lineares (NIELSEN, 2015). Embora tenham despertado bastante interesse na época, muitas pesquisas que se seguiram demonstraram que seus resultados não eram promissores. Muitos algoritmos de RL, por exemplo, quando combinados com aproximadores simples não garantiam uma convergência da solução (BAIRD, 1995; TESAURO, 1992; TSITSIKLIS; ROY, 1996). Por outro lado, os aproximadores lineares tornaram possível a resolução de problemas associados com a convergência em algoritmos com base em diferenças temporais - *temporal differences* (TD) (SUTTON, 1996; TSITSIKLIS, 1997; BRADTKE; BARTO, 1996; SCHOKNECHT; MERKE, 2003). Contudo, existe uma limitação no que refere-se ao seu desempenho, o qual é influenciado diretamente pela escolha de suas funções base (XU; GAO, 2008). Estes métodos podem ser considerados originários da psicologia, utilizando o conceito do reforço secundário (SUTTON, 1988).

Ainda a respeito do comportamento animal, pesquisas revelaram que através de

[5] Alan Mathison Turing foi um foi um matemático inglês, cientista da computação, lógico, criptanalista, filósofo e biólogo teórico (1912 - 1954).

[6] Marvin Minsky foi um cientista cognitivo norte-americano (1927 - 2016).

treinamentos, um reforço secundário pode ser associado a um reforço primário (recompensa ou punição) (PAVLOV, 1927). Após a realização de tal associação entre os reforços primário e secundário, o animal em treinamento pode vir a ser estimulado utilizando apenas o reforço secundário ao invés do reforço primário original. Um cão, por exemplo, pode ser treinado para realizar tarefas específicas, utilizando um alimento como recompensa e, associado a este, um determinado som. Após isso, o cão pode ser ensinado a aprender tarefas para as quais a recompensa é o próprio som, mesmo se este não vier acompanhado de um alimento (WILLIAMS, 2009b). Neste contexto, os métodos de TDs realizam uma propagação de informações importantes "para trás" em uma sequência de experiências, tal que as ações que levam ao sucesso comportamental são reforçadas, mesmo quando suas recompensas imediatas levam a um atraso significativo.

A reunião das ideias do TD, DP e o aprendizado por tentativa e erro, deu origem ao chamado algoritmo *Q-learning* (SUTTON; BARTO; WILLIAMS, 1992). A simplicidade e versatilidade desse algoritmo rapidamente o tornou um dos mais conhecidos e populares algoritmos da área do RL, principalmente nos últimos tempos. Além disso, o *Q-learning* pode ser encarado como um método de controle ótimo adaptativo direto, como demonstrado em trabalhos anteriores(SUTTON; BARTO; WILLIAMS, 1992).

Além do algoritmo *Q-learning*, destacam-se como métodos importantes na área do RL aqueles baseados na arquitetura ator-crítico - *actor-critic* (AC) (SUTTON; BARTO, 2018). Estes algoritmos têm uma estrutura de memória em separado, representando a política, independentemente da função valor. A estrutura que tem a função da política é denominada de ator, pois tem o objetivo de selecionar as ações a serem tomadas. No caso da função valor estimado, a estrutura que a representa denomina-se de crítico, pois tem o objetivo de criticar as ações escolhidas pelo ator (BARTO; SUTTON; ANDERSON, 1983). O processo de aprendizagem nesse caso depende da política: o crítico precisa aprender e "criticar" qualquer política que esteja sendo utilizada pela estrutura do ator.

1.1 O Aprendizado por Reforço Profundo

Os algoritmos de RL são uma importante classe de métodos da área do aprendizado de máquina - *machine learning* (ML). A ML tem se tornado cada vez mais popular, chamando a atenção de vários pesquisadores em diversas áreas distintas. O resultado disso é o grande número de recentes aplicações, podendo-se destacar a recuperação de dados multimídia, classificação, recomendação de vídeo, análise de redes sociais, e assim por diante. Neste contexto, uma subárea do ML, o "aprendizado profundo" - *deep learning* (DL) ou aprendizado de representação (DENG, 2014) tem fundamental importância nessas aplicações (HA et al., 2015; GOODFELLOW; BENGIO; COURVILLE, 2016; GOODFELLOW et al., 2014). O crescimento vertiginoso na disponibilidade de dados[7], bem como os grandes avanços recentes nas tecnologias de *hardware*

[7] *Big data*.

propiciaram o surgimento de novos estudos nessa área (POUYANFAR et al., 2018). A DL tem suas origens nas ANNs convencionais, superando-as de forma significativa. Recentes trabalhos na área do DL demonstraram resultados muito promissores em diferentes áreas, destacando-se o Processamento de Linguagem Natural - *natural language processing* (NLP), processamento de dados visuais, processamento de fala e áudio e muitas outras aplicações conhecidas (YAN et al., 2017; YAN et al., 2015; LACHAUX et al., 2020; MOON; L., 2020).

De forma geral, um algoritmo de ML tem sua eficiência relacionada com a qualidade da representação dos dados. Por esta razão, a má representação dos dados, quase sempre leva a um desempenho menor em comparação a uma boa representação dos dados. Portanto, o tratamento dos dados têm sido um importante pilar na pesquisa em ML, concentrando-se na construção de características a partir de dados "brutos". Além disso, esse trabalho de caracterização dos dados costuma ser muito específico, requerendo um esforço humano significativo (POUYANFAR et al., 2018). Por outro lado, os algoritmos DL realizam a extração de características de forma automática, permitindo dessa forma extrair dados discriminativos com menor conhecimento e esforço humano (NAJAFABADI et al., 2015). Por meio de arquiteturas de redes neurais em representações hierárquicas, os adeptos do ML têm feito grandes progressos na abordagem da maldição da dimensionalidade (BENGIO; COURVILLE; VINCENT, 2013).

Assim como em diversas outras áreas, o DL acelerou de maneira semelhante o progresso no ramo do RL, caracterizando-se pela utilização de algoritmos com DL dentro do RL. A este fato, deu-se a criação de um novo campo de conhecimento denominado de “aprendizagem por reforço profundo” - *deep reinforcement learning* (DRL) (ARULKUMARAN et al., 2017). O DRL permitiu abordar problemas de tomada de decisão que eram intratáveis com o RL, ou seja, problemas que possuíam configurações com estados e espaços de ação de alta dimensionalidade.

Atualmente, diversos algoritmos de DRL têm sido aplicados a uma ampla gama de problemas, como no caso da robótica, onde as leis de controle dos robôs são aprendidas diretamente das entradas de imagens de câmeras no mundo real (LEVINE et al., 2016b; LEVINE et al., 2016a). Além disso, algoritmos de DRL têm sido desenvolvidos para ter, cada vez mais, maior capacidade de aprendizado e, até mesmo a capacidade de “aprender a aprender” (WANG et al., 2017a). Neste caso, os algoritmos de DRL tornam-se capazes de generalizarem para ambientes complexos que nunca viram antes. Os últimos avanços produzidos na área do DL, cada vez mais contribuem para o desenvolvimento das técnicas do RL. Apesar da grande parte da teoria atual do RL apresentar limitações, principalmente pelo uso de métodos que utilizam aproximação de funções tabulares ou lineares, os desempenhos impressionantes de notáveis aplicações de DRL, devem-se em muito ao sucesso de aproximação de funções não lineares por ANNs com multicamadas (SUTTON; BARTO, 2018).

1.2 Controle Baseado na Aprendizagem Emocional

Assim como as recompensas ou punições são importantes na teoria do RL para modelar o processo de aprendizagem dos seres vivos, as emoções podem fornecer um modelo de aprendizado adaptativo muito eficaz, conhecido como o aprendizado baseado nas emoções - *emotional learning* (EL). O EL se destaca no sentido de que a avaliação emocional dos estímulos pode fornecer respostas mais rápidas e, garantir em determinadas situações, uma maior sobrevivência ao indivíduo. Devido a este e outros motivos, nota-se que existem diversos trabalhos e aplicações na área do EL (LOTFI; KHAZAEI; KHAZAEI, 2017; LOTFI; KHOSRAVI; NAHAVANDI, 2017), os quais possuem inspiração, principalmente, no funcionamento e arquiteturas do aprendizado no cérebro humano (LUNDAGARD; BALKENIUS, 2000; MIRHAJIANMOGHADAM; AKBARZADEH; LOT, 2016).

Por meio de ações definidas ou não, os seres humanos reagem às diversas situações do ambiente ao seu redor. Na maioria das vezes, tais ações são de cunho lógico, no entanto, podem ser influenciadas pelas emoções. Sem dúvidas, as emoções têm um papel vital na vida e, além disso, são um ativo valioso na sobrevivência e adaptação. Uma corrente bem difundida a respeito desse tema aborda que as emoções foram inseridas ao longo do processo evolutivo, funcionando como um mecanismo para a redução do tempo de reação aos estímulos externos. De certa forma, ao invés de utilizar a parte do cérebro responsável pelo raciocínio para o processamento de informações e, após isso, produzir um conjunto específico de ações, acarretando um certo tempo, a reação por emoção, por outro lado, seria muito mais rápida. Nesse sentido, uma analogia que pode ser feita a respeito das emoções é considera-lá como um piloto automático, desenvolvido após milhões de anos de evolução (GOODFELLOW; BENGIO; COURVILLE, 1997).

As emoções descrevem uma maneira de reação automática ao mundo de um modo inconsciente. A parte da estrutura cerebral responsável pela atividade emocional é denominada de sistema límbico. Todavia, esse sistema não é limitado ao controle das emoções, existe uma variedade de outras funções, como por exemplo na questão comportamental e motivacional, além de ter um impacto expressivo no processo de formação da memória. Tal sistema também promove a interconexão entre sinais emocionais e de raciocínio, levando a um estado de baixo estresse emocional (GOODFELLOW; BENGIO; COURVILLE, 1997; LAUTIN, 2001)

No que se refere ao sistema límbico, pode-se destacar como uma importante área o córtex frontal, especificamente o córtex orbitofrontal - *orbitofrontal cortex* (OC). A partir de resultados experimentais em seres humanos, obtidos em (VALENTIN; DICKINSON; O'DOHERTY, 2007), foi sugerido que o OC é um componente cerebral muito importante para o comportamento guiado por objetivos. Em (PADOA-SCHIOPPA; ASSAD, 2006), através de testes em macacos, apoiou-se a tese da contribuição do OC na codificação de valores do comportamento da escolha guiada. Além disso, outras revisões no âmbito da neuroeconomia contribuíram para o entendimento sobre como o cérebro toma decisões direcionadas a objetivos (RANGEL; CAMERER; MONTAGUE, 2008; RANGEL; HARE, 2010). No trabalho de (PEZZULO et

al., 2014), revisou-se a neurociência das sequências de ativações geradas internamente e, a partir disso, apresentou-se um modelo de como esses mecanismos podem ser componentes do planejamento baseado em modelos. Em (DAW; SHOHAMY, 2008), propôs-se que enquanto os sinais da dopamina se conectam bem ao comportamento habitual (livre de modelo - *model-free*), outros processos, no entanto, estão envolvidos no comportamento direcionado por objetivo (baseados em modelo - *model-based*). Dados experimentais de (BROMBERG-MARTIN et al., 2010) indicaram que os sinais de dopamina contêm certas informações pertinentes às condições de escolha na forma *model-free* e *model-based*. Além disso, o trabalho de (DOLL; SIMON; DAW, 2012) argumentou que pode não haver uma separação evidente no cérebro entre os mecanismos que mantêm a aprendizagem e a escolha de forma *model-based* ou *model-free*. De forma geral, as soluções geradas pela natureza para solucionar os problemas de adaptação ambiental de todas as espécies vivas têm sido objeto de extenso estudo e análise.

Em resumo, o sistema emocional pode ser encarado como um elemento importante na robustez e capacidade da adaptação. Por este motivo, atualmente, várias aplicações na área da inteligência computacional e de sistemas de controle estão explorando o processo de aprendizagem e a resposta emocional. A principal função da emoção é avaliar os estímulos e focar a atenção do sistema sobre os sinais que mais contribuem para alcançar os objetivos deste sistema. Neste caso, ao invés de desperdiçar os recursos em todos os estímulos sensoriais, a avaliação emocional pode ajudar a concentrar-se nos estímulos mais relevantes no processo decisório. A respeito disso, o trabalho de (CESAR et al., 2017) realizou um estudo da performance de um controle baseado no aprendizado emocional do cérebro - *brain emotional learning* (BEL), aplicando-o a um amortecedor magnético.

Desde o surgimento deste modelo de controlador, diversas aplicações foram propostas na área da engenharia de controle. Neste âmbito, o trabalho de (JAFARI et al., 2017) demonstrou a utilização de um tipo de controlador biologicamente inspirado, cuja premissa se baseia no BEL, denominado de controle inteligente baseado no aprendizado emocional do cérebro - *brain emotional learning based intelligent controller* (BELBIC). O objetivo deste controlador era realizar o rastreamento inteligente de aeronaves não tripuladas na presença de incertezas nas dinâmicas e perturbações do sistema. O trabalho de (KHORASHADIZADEH; MAHDIAN, 2016), utilizou o BELBIC para realizar o controle de tensão em um conversor boost DC-DC. Por outro lado, o trabalho de (RAHMAN et al., 2008) foi o primeiro a propor a utilização do BELBIC em um motor síncrono de ímã permanente. Em (SADEGHI; DARYABEIGI, 2014), propôs-se um modelo novo e simples para o acionamento de um motor síncrono a imã permanente - *interior permanent magnet synchronous motor* (IPMSM) com base em BELBIC, capaz de realizar o controle da velocidade sem o uso de controladores *proportional-integer* (PI) convencionais e, além disso, ser independente dos parâmetros do motor. Em (YI, 2015), foi apresentado um controle robusto de modo deslizante bio-inspirado aplicado a um manipulador robótico com incertezas de rastreamento, cuja a estratégia se baseia no BELBIC. No caso do trabalho de (SADEGHIEH; ROSHANIAN; NAJA, 2012), realizou-se um rastreamento da posição para

servomotor em um sistema eletro-hidráulico, baseando-se em um controle com BELBIC. Em (JAMALI et al., 2008), propôs-se a utilização do BELBIC para o controle do posicionamento de um guindaste. Na área de sistemas de potência, o trabalho de (JAFARI et al., 2013) associou um controlador BELBIC com um controlador do tipo PI, aplicando-os a um dispositivo de controle de fluxo de potência, tendo como principal objetivo a melhora da estabilidade transitória. Em (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004), utilizou-se com sucesso um controlador BELBIC para controle de processos lineares. Por outro lado, referindo-se a sistemas não-lineares, o trabalho de (LUCAS; RASHIDI; ABDI, 2004; RAHMAN et al., 2008) aplicou o BELBIC para o controle de uma máquina síncrona de imã permanente e um regulador de tensão automático. Essas e outras aplicações têm demonstrado, cada vez mais, a eficácia deste tipo de controlador em áreas distintas, abrangendo os mais variados tipos de sistemas de controle.

Além das próprias aplicações do controlador emocional (BELBIC), pode-se ainda associá-lo a outros tipos de controladores, obtendo assim novas configurações para esse esquema de controle. Neste contexto, uma aplicação possível foi a associação deste tipo de controlador com as ANNs. Esta arquitetura trouxe novos benefícios, pois esse esquema de controle não utiliza somente os erros de saída da rede neural para ajustar os pesos entre as próprias conexões da rede, mas também faz uso dos benefícios da saída emocional, utilizada como avaliador do desempenho geral para ajustar seus parâmetros (ZHOU et al., 2015; ZHOU et al., 2017).

No geral, o controlador BELBIC pode ser altamente eficaz e confiável em aplicações que requeiram alto desempenho, desde que seja possível utilizar um algoritmo para o ajuste ou controle confiável de suas necessidades paramétricas. No entanto, apesar dos grandes avanços na área do controle emocional, ainda existe um longo caminho a ser percorrido no ambiente de estudos deste tipo de controlador.

1.3 Motivações

Na literatura, a formulação matemática do sistema de controle baseado no aprendizado emocional do cérebro se baseia, principalmente, em modelos simplistas do sistema límbico (MORéN, 2002; MORéN; BALKENIUS, 2000). Esses modelos são feitos com base em alguns requisitos necessários, como por exemplo, os estímulos sensorial - *sensory stimulus* (S) e emocional - *emotional stimulus* ou recompensa - *reward* (*R*). Na engenharia de controle, o projeto do controlador BELBIC passa obrigatoriamente pela definição de ambos os sinais de estímulos *S* e *R* (LOTFI; REZAEE, 2018). Estes estímulos são modelados tal que possam representar os objetivos e estados dos sistemas dinâmicos em diferentes aplicações. Além disso, associado aos critérios de desempenho do sistema, os objetivos de controle, está o estímulo *R*. Por outro lado, o estímulo *S* está relacionado à velocidade da resposta do controle, além de outras considerações de engenharia de controle (LOTFI; REZAEE, 2018). Tais sinais podem ser definidos de diversas maneiras, utilizando-se por exemplo as variáveis disponíveis na malha de controle do sistema, a exemplo do erro, derivada do erro, integral do erro, *feedbacks* dos

sinais de controle e saída da planta, entre outras possibilidades. A partir de tais considerações, observa-se que a definição dos estímulos para o controlador emocional, geralmente, é diferente para cada tipo de aplicação, dependendo dos objetivos do projetista. Esta característica do controlador BELBIC o torna um tipo de controle não generalista (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004).

Diversos trabalhos relacionados ao desenvolvimento do controlador emocional se utilizaram da mesma premissa, atribuindo-se diferentes variáveis e ganhos para compor os sinais S e R em aplicações específicas. No trabalho de (SHARBAFI; LUCAS; DANESHVAR, 2010), por exemplo, o sinal R é formulado por uma arquitetura do tipo proporcional integral derivativa - *proportional integral derivative* (PID), por outro lado, o sinal S foi composto apenas pela derivada do erro com um ganho específico. Em (DEHKORDI et al., 2011; MARKADEH et al., 2011), pesos foram definidos de forma específica para o S e R, tal que controlasse a velocidade da resposta deste controlador. No caso do trabalho de (DEHKORDI et al., 2011), o R foi definido como um PID e o S apenas por um ganho proporcional ao erro. (MARKADEH et al., 2011) definiram R também como um PID, no entanto, S foi definido como uma combinação dos módulos do erro e *feedback* da planta.

Uma vez que os estímulos S e R são de fundamental importância na dinâmica do controlador emocional, faz-se necessária a correta compreensão dos efeitos de ambos os estímulos no resultado final de controle. Além disso, as variáveis da malha de controle do sistema que, porventura possam compor os estímulos, precisam ser devidamente manipuladas.

De fato, não existe na literatura uma metodologia exata para a formulação dos estímulos S e R do controlador emocional. Neste sentido, a modelagem destes sinais de estímulos tornam-se um processo iterativo de ajustes, envolvendo diversos testes e simulações para determinar os melhores valores atribuídos aos ganhos e, principalmente, quais as variáveis da malha de controle devem estar ou não envolvidas. Por esta razão, evidenciada em trabalhos relacionados ao tema, nota-se a dificuldade na formulação adequada destes estímulos, o que pode acarretar a deficiência de desempenho deste tipo de controlador e, até mesmo torná-lo inviável na prática.

1.4 Objetivos

De forma geral, a proposta do presente trabalho se concentra em desenvolver e aplicar uma metodologia para a construção dos estímulos S e R que compõem o controlador emocional. Uma vez que estes sinais de estímulos dependem dos requisitos do projetista e da dinâmica do processo a ser controlado, propõe-se utilizar uma arquitetura com base em aprendizado adaptativo, especificamente o DRL. A proposta é unir o controlador BELBIC ao estado da arte das técnicas de DRL.

A motivação para a utilização do DRL se baseia em sua capacidade de obter leis de controle não lineares diretamente a partir de iterações com o sistema dinâmico. Uma vez que

os algoritmos pertencentes a esta classe de métodos se utilizam de ANNs em suas arquiteturas, faz-se possível realizar a aproximação de funções não lineares. Portanto, desta maneira é possível se obter leis de controle mais eficazes para cada um dos estímulos, S e R, tornando possível a realização de determinada tarefa operacional de modo eficiente por parte do controlar emocional.

Devido à falta de uma metodologia específica no que diz respeito à composição dos sinais dos estímulos S e R, considerando as diferentes situações de aplicações do controlador emocional, o presente trabalho se utiliza do DRL com o objetivo de fornecer uma característica de generalização para este tipo de controlador. Neste sentido, pode ser útil uma abordagem diferente dos métodos tradicionais de controle linear para a construção de tais sinais de estímulos. Uma vez que, além da possibilidade de descrever as leis de controle dos estímulos através de equações dinâmicas não lineares, pode-se obter diretamente estes sinais. Tal situação é possível apenas observando as características dinâmicas dos sistemas controlados em questão.

Além disso, pretende-se demonstrar com este trabalho a possibilidade da utilização desta metodologia em processos industriais de forma prática. A partir da concepção, análise e aplicação prática do controle emocional associado ao DRL, busca-se tonar esta arquitetura de controlador mais abrangente e escalável no mercado.

1.5 Contribuições

O presente trabalho apresenta contribuições que envolvem diversas áreas do conhecimento que, a princípio são de grande relevância prática. Em primeiro lugar, apresenta-se uma revisão da literatura do controlador emocional e suas aplicações na engenharia de controle. Além disso, revisa-se também a literatura do DRL e suas implicações na teoria do controle.

A principal contribuição do trabalho é a apresentação de uma metodologia para a construção das arquiteturas dos sinais de estímulos S e R que compõem o controlador emocional, utilizando-se como base as recentes técnicas de DRL. O DRL permite extrair padrões característicos das dinâmicas dos sistemas que, porventura possam a vir a ter uma alta dimensionalidade e dinâmicas possivelmente não-lineares, como é o caso da maioria dos problemas que envolvem os sistemas dinâmicos do mundo real. Dessa forma, é possível obter sinais de estímulos condizentes com o sistema a ser controlado, pois é possível produzir uma lei de controle adaptável para cada problema específico. Assim sendo, não há a princípio a necessidade de um grande conhecimento humano a respeito do processo em questão. Portanto, espera-se que a lei de controle dos estímulos resulte em um controlador emocional mais robusto.

Além da proposta metodológica para a construção dos estímulos do controlador emocional, propõe-se ainda uma modificação nas características de aprendizado deste tipo de controlador. A partir de modificações na forma dos ganhos do aprendizado dos submódulos do BELBIC, pode-se obter uma melhora da performance deste controlador, referindo-se a estabilidade e velocidade de resposta do sinal de controle.

No que diz respeito ao desenvolvimento da proposta deste trabalho, destaca-se a formulação de uma metodologia para a construção, teste e comissionamento do controlador emocional por DRL via plataformas de programação distintas. A partir da utilização de diferentes algoritmos de DRL, associados com sistemas dinâmicos distintos, demonstra-se uma forma prática para a construção deste tipo de controlador via um ambiente de comunicação em tempo real.

Por fim, o trabalho demonstra a utilização prática do controlador proposto em um ambiente que envolve sistemas industriais. Nesse sentido, o trabalho delineia a implementação do controlador em um *hardware* dedicado e, além disso, apresenta o modelo de sua interação com a planta do processo industrial via sistema de automação. Essa implementação é elaborada tal que se faça possível replicá-la para outros ambientes dinâmicos.

1.6 Organização do trabalho

De forma a demonstrar a metodologia desenvolvida, o presente trabalho apresenta a seguinte estrutura organizacional:

- **Capítulo 2 -** Realiza uma revisão da teoria do RL. Nesse caso, apresentam-se alguns conceitos importantes a respeito deste tema, demonstrando sua metodologia de funcionamento e implicações em sistemas de controle. Além disso, o capítulo realiza uma abordagem acerca da recente área do DRL.

- **Capítulo 3 -** Apresenta uma revisão do mecanismo de funcionamento do aprendizado emocional do cérebro, destacando seus principais componentes e respectivas funções. Ainda neste capítulo, trata-se da modelagem computacional do controlador emocional e, a respectiva importância dos sinais de estímulos para o mesmo.

- **Capítulo 4 -** É proposta a metodologia do trabalho. Neste capítulo, destacam-se todas as ferramentas necessárias a realização do trabalho, bem como os parâmetros importantes na elaboração da proposta. Além disso, demonstra-se como ponto central o modelo dos estímulos do controlador emocional através do DRL. Por fim, abordam-se alguns pontos importantes a respeito da estabilidade do controlador proposto.

- **Capítulo 5 -** Apresentam-se os resultados e discussões gerais do trabalho. Neste capítulo, demonstra-se o projeto do controlador, assim como suas características de comissionamento. Ainda neste capítulo, todas as etapas de treinamento, simulação e análise dos resultados do controlador proposto são realizadas. Além disso, trata-se a proposta da modificação do aprendizado do controlador emocional.

- **Capítulo 6 -** Apresenta as conclusões e os possíveis trabalhos futuros referentes aos temas abordados neste trabalho.

1.7 Trabalhos publicados

- R. B. Aquino, L. F. A. Cordeiro, D. C. Marques, J. L. Silva, C. M. Bandeira and A. L. M. M. Andrade, "An emotional controller PLC implementation for an industrial fan system,"2016 International Joint Conference on Neural Networks (IJCNN), Vancouver, BC, 2016, pp. 3889-3895.
- J. L. Silva, D. C. Marques, R. B. Aquino, O. N. Neto, "PLC-based on Fuzzy Logic Control with Metaheuristic Tuning", Studies in Informatics and Control - ICI Bucharest.

2 SISTEMAS DE APRENDIZADO POR REFORÇO

As técnicas inspiradas a partir da natureza, principalmente no que diz respeito à forma de aprendizado dos indivíduos, como as ANNs e o RL, auxiliam no desenvolvimento de soluções para os mais diversos tipos de problemas, têm apresentado resultados promissores.

2.1 Introdução

O desenvolvimento de controladores que produzam ações ótimas para o controle do comportamento dinâmico de um sistema é de muita importância, até mesmo crucial em diversas áreas, como no caso da robótica, dos processos industriais e sistemas de voos espaciais. Desde o início dos tempos, a construção de controladores automáticos para mecanismos de todos os tipos têm sido um grande desafio para cientistas e engenheiros. Muitos dos principais esforços de pesquisas referentes a este tema, deram-se com o intuito de abordar as questões teóricas levantadas por tais situações e, além disso, fornecer métodos práticos para a construção de controladores eficientes.

2.2 O aprendizado de máquina

Nas últimas décadas, presencia-se uma revolução no trabalho em AI, conjuntamente em conteúdo e metodologia. A prática atual se utiliza das teorias existentes como base teórica, diferentemente do que acontecia antes, onde eram propostas teorias inteiramente novas. Além disso, as afirmações são fundamentadas em teoremas rigorosos ou na evidência experimental rígida, em vez de utilizar como base a própria intuição. Nesse caso, o foco dos estudos em AI concentra-se na relevância de aplicações reais em vez de exemplos com brinquedos (RUSSELL; NORVIG, 2010).

Do mercado de trabalho às indústrias, passando pelo comportamento da sociedade no que diz respeito à forma de consumo de bens e serviços, a AI está presente em uma parcela significativa desses sistemas.

No que diz respeito à área de estudos referentes a AI, existem diferentes formas de "aprender"[1] um determinado conhecimento. Esta situação é abordada computacionalmente a partir da classe dos algoritmos conhecidos de ML (Figura 1).

[1] Em AI pode ser considerado como a capacidade ou habilidade de um máquina em realizar uma determinada tarefa.

Figura 1 – Aplicações dos algoritmos de aprendizagem de máquina.

Fonte: Próprio autor.

De todas as formas conhecidas de ML, o RL pode ser encarado como o mais próximo do tipo de aprendizado que humanos e outros animais fazem e, grande parte dos principais algoritmos de RL foram originalmente inspirados em sistemas de aprendizado biológico. A contribuição desse tipo de aprendizado se deu, tanto através de um modelo psicológico de aprendizado animal que combina melhor com alguns dados empíricos, como também por meio de um modelo do sistema de recompensa do cérebro.

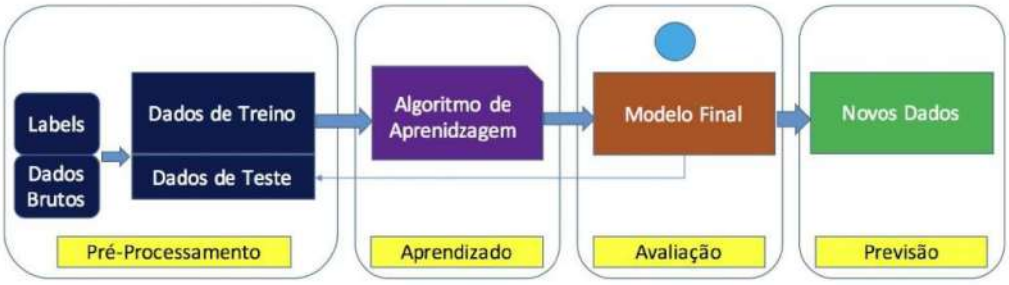
A aprendizagem por meio do reforço é considerada diferente daquela conhecida como a aprendizagem supervisionada, a qual é tema de grande parte das atuais pesquisas referentes à área de ML. O aprendizado supervisionado - *supervised learning* é realizado a partir de um conjunto de treinamento com exemplos rotulados, os quais são fornecidos por um supervisor externo qualificado. Neste caso, pode-se considerar cada exemplo como uma descrição de uma dada situação, juntamente com uma especificação - o rótulo - da ação correta, a qual o sistema deverá tomar para essa situação, como por exemplo, identificar uma categoria à qual a situação pertence. Alguns dos principais algoritmos dessa classe de aprendizado são: regressões e classificações lineares, árvores de decisão, análise de componentes principais, *K-means*, entre outros (RUSSELL; NORVIG, 2010; MOHRI; ROSTAMIZADEH; TALWALKAR, 2012; VAPNIK, 2000).

O objetivo do aprendizado supervisionado pode ser considerado como a generalização ou extrapolação das respostas do sistema, tal que ele atue corretamente em situações que não estão

presentes no conjunto de treinamento fornecido. Sem dúvidas, esse é um tipo muito importante de aprendizado, no entanto, sozinho não é adequado para aprender com a interação no ambiente. Em problemas interativos, como no caso dos sistemas de controle, muitas vezes não é possível se obter exemplos de comportamentos desejados que sejam corretos e, representativos de todas as possíveis situações em que o agente tem que agir. Em um território ainda inexplorado, no qual se esperaria que aprender fosse o mais benéfico, um agente deve ser capaz de aprender a partir de sua própria experiência.

Além do aprendizado supervisionado, o RL também é considerado distinto de uma outra classe de algoritmos de ML, denominado de aprendizado não supervisionado - *unsupervised learning*. Esse tipo de aprendizado, basicamente, baseia-se em encontrar estruturas ocultas em um coleções de dados, os quais porventura não são rotulados. Os termos aprendizagem supervisionada e aprendizagem não supervisionada parecem classificar exaustivamente os paradigmas de ML, mas não o fazem. De certo modo, talvez possa haver uma "tentação" em relacionar o RL como um tipo de aprendizado não supervisionado, pois ele não se baseia em exemplos de um comportamento correto. Na realidade o RL busca maximizar um sinal de recompensa em vez de tentar encontrar estruturas ocultas a partir dos dados fornecidos. Por outro lado, a descoberta de possíveis estruturas implícitas por meio da experiência de um agente pode, sem dúvidas, ser muito útil no RL, contudo, por si só não aborda o problema do RL. Neste sentido, considera-se o RL como um terceiro paradigma de ML, juntamente com o aprendizado supervisionado e o aprendizado não supervisionado e talvez outros paradigmas. De forma geral, os algoritmos de ML possuem uma estrutura semelhante (Figura 2).

Figura 2 – Forma geral dos algoritmos de aprendizado de máquina.



Fonte: adaptado de (SIEMENS, 2019).

2.3 A teoria da aprendizagem por reforço

O RL pode ser encarado como uma abordagem computacional que busca entender e automatizar o aprendizado direcionado por objetivos, além de abordar o processo da tomada

de decisões. Esse tipo de aprendizado, distingue-se de outras abordagens computacionais, principalmente por sua grande ênfase na aprendizagem por meio de um agente que interage diretamente com seu ambiente, sem exigir uma supervisão, exemplos ou modelos completos do ambiente (SUTTON; BARTO, 2018).

Algumas literaturas consideram o RL como o primeiro campo a abordar diretamente os problemas computacionais que, porventura surgem ao aprender a partir da interação com um ambiente, a fim de alcançar objetivos de longo prazo (SUTTON; BARTO, 2018).

2.3.1 Processo de decisões sequenciais

Na maioria dos problemas que envolvem situações práticas, as decisões não são tomadas isoladamente, apenas admitindo o tempo e estado presentes, mas também suas consequências futuras. Na prática, as decisões precisam ser tomadas de forma sequencial em diferentes instantes de tempo e, até mesmo diferentes pontos no espaço. Esses tipos de problemas, envolvendo decisões sequenciais, denominam-se problemas de tomada de decisões sequenciais. Uma formalização clássica do processo de tomada de decisões sequenciais surgiu a partir dos trabalhos de Markov[2], denominada de processo de decisão markoviano - *Markov decision process* (MDP). Nos processos de MDP, ações influenciam não apenas as situações imediatas, mas também os estados subsequentes (PUTERMAN, 1994).

Os MDPs são na realidade, uma formulação matemática idealizada do problema de RL, no qual podem ser feitas afirmações teóricas precisas. A pretensão dos MDPs é abordar diretamente o problema de aprendizado por meio da interação, de forma a alcançar um determinado objetivo proposto. Contudo, não diferente de outros casos da área de AI, existe sempre uma preocupação entre o grau de aplicabilidade da proposta e sua tratativa matemática.

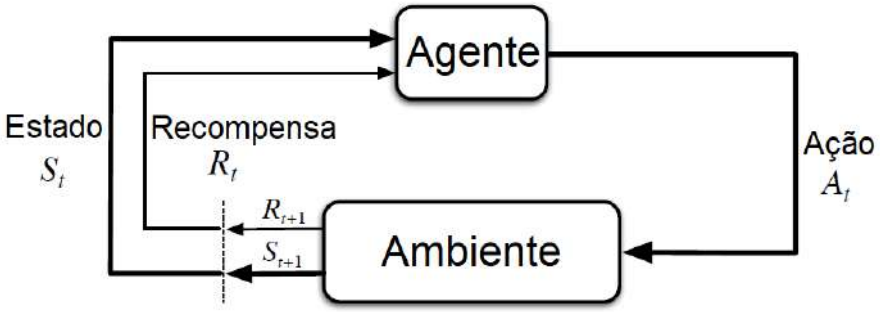
No contexto dos MDPs, existe uma estrutura denominada de *agente*, responsável por "aprender" e tomar decisões. A outra estrutura presente nesse modelo é denominada de *ambiente*, basicamente, é tudo fora do *agente*, a coisa com a qual interage-se. O *ambiente* e o *agente* interagem continuamente entre si. Nesse sentido, o *agente* seleciona as ações e o *ambiente* responde a essas ações e apresenta novas condições ao *agente*.

No que diz respeito aos MDPs, a Figura 3 ilustra o processo de interação entre o agente e o ambiente.

---

[2] Andrei Andreyevich Markov foi um matemático russo (1856 - 1922).

Figura 3 – Agente e ambiente em um processo de decisão markoviano.



Fonte: adaptado de (SUTTON; BARTO, 2018).

No MDP, o agente interage com o ambiente por meio da seleção de ações, $A_t \in \mathcal{A}(s)$, em uma sequência discreta de tempo *t*, recebendo deste alguma representação do seu estado, $S_t \in \mathcal{S}$. Um passo de tempo após a ação, como resultado desta, o agente recebe uma recompensa, $R_{t+1} \in \mathcal{R}$, e o ambiente encontra-se em um novo estado S_{t+1}. A dinâmica do MDP é ditada por:

$$p(s', r|s, a) \doteq P_r\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}, \tag{1}$$

para todo $s', s \in \mathcal{S}$, $r \in \mathcal{R}$ e $a \in \mathcal{A}(s)$. A função *p* estabelece a dinâmica do MDP, demonstrando que existe uma probabilidade de s'e r ocorrerem em um instante de tempo t, dados valores particulares de estados (s) e ações precedentes (a). A partir dessa definição, torna-se possível calcular outras situações para o ambiente, como a *probabilidade de transição de estados*,

$$p(s'|s, a) \doteq P_r\{S_t = s'|S_{t-1} = s, A_{t-1} = a\} = \sum_{r\in\mathcal{R}} p(s', r|s, a), \tag{2}$$

as recompensas esperadas para um par *estado-ação*,

$$r(s, a) \doteq E[R_{t+1}|S_t = s, A_t = a] = \sum_{r\in\mathcal{R}} r \sum_{s'\in\mathcal{S}} p(s', r|s, a), \tag{3}$$

e as recompensas esperadas para a tripla relação *estado-ação-próximo estado*,

$$r(s, a, s') \doteq E[R_{t+1}|S_t = s, A_t = a, S_{t+1} = s'] = \sum_{r\in\mathcal{R}} r\frac{p(s', r|s, a)}{p(s'|s, a)}. \tag{4}$$

A estrutura do MDP pode ser considerada bastante abstrata e flexível, podendo ser aplicada a diversos tipos de problemas em diferentes formas. As etapas de tempo não precisam necessariamente ser intervalos fixos de tempo real, podendo ser etapas sucessivas arbitrárias de tomada de decisão e atuação. As ações podem englobar diversas situações, por exemplo, controles de baixo nível, como as tensões aplicadas aos motores de um braço robótico, ou decisões de alto nível, como por exemplo se deve ou não almoçar ou ir para a pós-graduação.

De maneira semelhante, os estados podem assumir uma ampla variedade de formas, podendo ser completamente determinados por sensações de baixo nível, por exemplo, leituras diretas de sensores, ou podem ser mais de alto nível e abstratas, como descrições simbólicas de objetos em uma sala (SUTTON; BARTO, 2018).

2.3.2 Políticas e recompensas

A parte central de um agente de RL é denominada de *política*. Essa política pode ser definida como a forma com que um agente comporta-se em um dado momento no ambiente ao qual está inserido. De forma geral, a política é o mapeamento dos estados do ambiente que são percebidos pelo agente e as consequentes ações que devem ser tomadas nestas situações. Na área da psicologia, essa situação pode ser comparada ao que denomina-se de "associações estímulos-respostas[3]". No geral, tais políticas podem ser do tipo estocásticas, determinando assim, probabilidades para cada ação. A política pode então ser definida como

$$\pi = \{h_0, h_1, h_2, h_3, h_4, ...\}, \tag{5}$$

sendo que h_t é definido como o mapeamento $h : \mathcal{S}$ x $\mathcal{A} \longmapsto [0, 1]$, fornecendo a probabilidade condicional para cada estado s e ação a,

$$h_t(a|s) = P_r\{A_t = a|S_t = s\}, a \in \mathcal{A}(s), \forall s \in \mathcal{S}. \tag{6}$$

Durante o processo de aprendizagem, em cada instante de tempo, o ambiente envia ao agente de RL um valor específico denominado de *recompensa*. Todos os problemas de RL têm seus objetivos centrados em tais sinais de recompensa, de modo que um agente precisa executar ações específicas que, ao longo do tempo, promovam a maximização desse sinal recebido. De forma simples, em termos de eventos, o sinal de recompensa reflete o que é bom ou ruim para um agente. Tal situação é análoga no campo da psicologia, relativamente às experiências com dor ou prazer associadas. Este sinal funciona como uma situação primária para realizar uma modificação na política do agente. Por exemplo, caso uma determinada ação seja executada pelo agente e, seguindo-se a esta, um sinal de recompensa recebido seja "prazeroso", haverá a tendência de seguir tal política no futuro. Do contrário, caso o sinal de recompensa recebido seja no sentido de "penalidade", tal política poderá ser alterada de forma a selecionar uma outra ação nessa situação no futuro.

2.3.3 Funções de retorno

Em um problema de MDP, o agente precisa escolher a melhor política que ocasione a maximização das recompensas recebidas ao longo de uma determinada sequência de eventos.

[3] Comportamento baseado na interação, o qual a resposta é emanada imediatamente após a apresentação de um estímulo.

Nesse sentido, deseja-se maximizar o *retorno esperado*, onde esse valor é definido como G_t, uma função das sequências de recompensas recebidas (SUTTON; BARTO, 2018). A formulação da função retorno é da seguinte forma:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + ... + R_T, \tag{7}$$

onde *T* é o último passo. Essa aproximação tem sentido quando admite-se a noção natural de tempo final, ou seja, quando a forma de interação entre agente-ambiente é do tipo episódica, tal como a maioria dos jogos de vídeo game. Contudo, essa interação agente-ambiente, em muitos casos, não acontece de forma episódica, mas de forma contínua, tornando (7) inviável. Na forma contínua, o *T* torna-se infinito, logo o valor do retorno esperado, o qual espera-se maximizar, poderá ser também infinito.

De maneira a tratar tal situação limitadora, adapta-se a equação do retorno esperado para situações contínuas pela adição de um *fator de desconto* γ. Neste caso, (7) se torna:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{8}$$

onde γ é parametrizado entre 0 e 1. Esse fator traduz o valor de recompensas futuras no tempo presente. Por exemplo, caso uma recompensa seja recebida em *k* passos à frente, seu valor será γ^{k-1} vezes o que ela valeria caso fosse recebida imediatamente. Por essa razão, quanto mais próximo de 0 for o valor do fator de desconto, maior será a importância que o agente dará para recompensas imediatas. De forma equivalente, à medida que γ aproxima-se de 1, recompensas futuras são consideradas de maior relevância pelo agente.

2.3.4 Funções valor

O sinal de recompensa está associado a situações em que ocorrem avaliação imediata da ação, no entanto, quando deseja-se referir àquelas avaliações de longo prazo, pode-se recorrer à chamada *função valor*. As funções valores podem ser estimadas a partir das sequências de observações que um agente realiza ao longo da vida. Por outro lado, as recompensas são obtidas diretamente pelo ambiente. Caso não houvesse recompensas, não haveria sentido em abordar a função valor, pois o maior propósito de sua estimativa é a obtenção de maiores recompensas.

A seleção e a respectiva avaliação das decisões são feitas, principalmente, baseando-se na função valor. Neste sentido, busca-se realizar ações que acarretem estados de maior valor, e não em maior recompensa, pois tais ações proporcionam maior recompensa a longo prazo. Todavia, a dificuldade presente é muito maior na determinação das funções valores do que na determinação das recompensas.

A função valor pode ser de dois tipos, dependendo de qual critério de avaliação está sendo proposto para avaliar o retorno esperado. Quando refere-se apenas ao quão bom é estar em

um determinado estado, utiliza-se a *função valor estado*. A função valor estado pode ser definida como a quantidade total de recompensas futuras que um agente espera acumular a partir desse estado. Esse valor indica a conveniência a longo prazo do estado, levando-se em consideração os estados que provavelmente sucederão a este, bem como as respectivas recompensas disponíveis nesses estados. Por outro lado, após avaliar quão boa é uma ação específica em um determinado estado, associa-se isto a uma *função valor estado-ação* (SUTTON; BARTO, 2018). Ambas as funções são definidas em torno de uma política específica π, pois as recompensas esperadas no futuro dependem das ações realizadas no presente.

Formalmente, à função valor de um estado *s* com uma política π, denomina-se $\upsilon_\pi(s)$. Este é o retorno esperado, iniciando-se no estado *s* e seguindo a política π. Em MDP, $\upsilon_\pi(s)$ é definido como:

$$\upsilon_\pi(s) \doteq E_\pi[G_t|S_t = s] = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s], \tag{9}$$

onde $E_\pi[\cdot]$ é o valor esperado de uma variável aleatória, admitindo que um agente siga a política π. De forma semelhante, a função valor estado-ação é definida tomando uma determinada ação *a* em um estado *s* com uma política π, de modo que:

$$q_\pi(s, a) \doteq E_\pi[G_t|S_t = s, A_t = a] = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s, A_t = a]. \tag{10}$$

Nos problemas que envolvem tomadas de decisões sequenciais, essas funções apresentam grande importância, pois podem ser reescritas de maneira a satisfazer a relação de recursividade[4] para trás no tempo. Além disso, as funções $\upsilon_\pi(s)$ e $q_\pi(s, a)$ podem ser estimadas através da experiência. Caso um agente siga uma política π e mantenha uma média, para cada estado encontrado, do atual retorno que seguiu-se a este estado, então a média irá convergir para $\upsilon_\pi(s)$. Tal situação também ocorre semelhantemente para a função q_π. Este tipo de estimação de valor é baseada nos algoritmos conhecidos como *métodos de Monte Carlo* (MC), pois envolvem uma média de muitas amostras aleatórias do atual valor do retorno (SUTTON; BARTO, 2018).

De modo a ilustrar a relação do MDP com o RL, apresenta-se a Figura 4. Nesse exemplo, o círculo representa um estado s e o círculo menor preenchido representa uma ação a. O valor de um determinado estado depende dos valores das ações possíveis nesse estado, ponderadas pela probabilidade de cada uma delas acontecer com base na política atual. A relação entre υ_π e $q_\pi(s, a)$ pode ser definida como

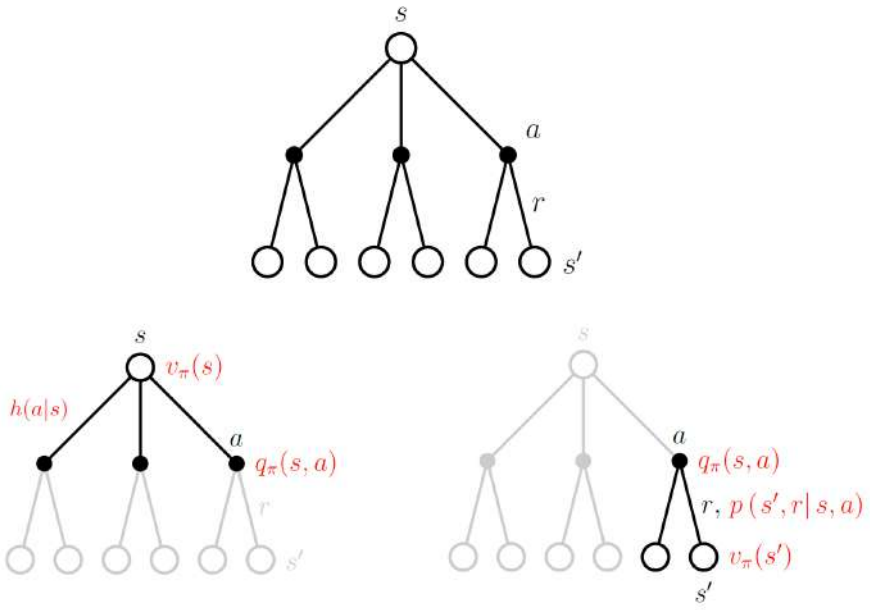
$$\upsilon_\pi(s) = \sum_a h(a|s) q_\pi(s, a). \tag{11}$$

[4] Processo de repetição de um objeto a partir de um jeito similar ao que já fora mostrado.

No caso de uma ação, seu valor está relacionado com a próxima recompensa esperada e com o respectivo somatório das recompensas remanescentes esperadas, ponderadas pela probabilidade de cada transição. O somatório das recompensas esperadas é caracterizado como a função valor estado para o seguinte estado s'. A relação entre $q_\pi(s,a)$ e $v_\pi(s')$, pode ser definida como

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]. \tag{12}$$

Figura 4 – Diagrama de decisões em RL.



Fonte: adaptado de (TEIXEIRA, 2016).

Uma das propriedades fundamentais das funções de valor, usadas ao longo do RL e da DP, é que elas satisfazem relações recursivas semelhantes àquelas estabelecidas na equação do retorno (8). Para qualquer valor de política π, bem como qualquer valor de estado s, uma condição de consistência é mantida entre o valor de s e o valor de seus possíveis estados sucessores, definida como

$$
\begin{aligned}
\upsilon_\pi(s) &\doteq E_\pi[G_t|S_t = s] = E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s] \\
&= E_\pi[R_{t+1} + \gamma \sum_{k=0}^{\infty} \gamma^k R_{t+k+2}|S_t = s] \\
&= \sum_a h(a|s) \sum_{s'} \sum_r p(s', r|s, a)[r + \gamma E_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+2}|S_{t+1} = s']], \\
&= \sum_a h(a|s) \sum_{s'r} p(s', r|s, a)[r + \gamma \upsilon_\pi(s')],
\end{aligned}
$$

$$
\upsilon_\pi(s) = \sum_a h(a|s) \sum_{s'r} p(s', r|s, a)[r + \gamma \upsilon_\pi(s')], \tag{13}
$$

a qual expressa o valor do estado e os valores de seus estados sucessores. Semelhantemente, existe relação análoga para função valor estado-ação q_π, definida como

$$
q_\pi(s, a) = \sum_{s'r} p(s', r|s, a)[r + \gamma \sum_{a'} h(a'|s') q_\pi(s', a')]. \tag{14}
$$

As equações (11) e (12) são formalmente conhecidas como as *equações de Bellman*[5] para a função valor estado e função valor estado-ação, respectivamente (BELLMAN, 1957).

2.3.5 Otimalidade de Bellman

Em RL a resolução de uma tarefa significa, a grosso modo, encontrar uma política que venha a alcançar uma grande recompensa a longo prazo. Uma política π é dita melhor ou igual à política π' se o seu retorno esperado é maior ou igual ao retorno esperado devido a π' para todos os estados. Em outras palavras, se $\pi \geq \pi'$ se e somente se $\upsilon_\pi \geq \upsilon_{\pi'} \forall s \in \mathcal{S}$. Existe pelo menos uma política que é melhor ou até mesmo igual a todas as outras políticas. Tal política é conhecida como uma política ótima (SUTTON; BARTO, 2018). Nesse caso, a função valor estado é conhecida como *função valor estado ótima*, definida da seguinte forma

$$
\upsilon^*(s) \doteq \max \upsilon_\pi(s) \qquad \forall s \in \mathcal{S}. \tag{15}
$$

No caso da função valor estado-ação com política ótima, essa relação torna-se

$$
q^*(s, a) \doteq \max q_\pi(s, a) \qquad \forall s \in \mathcal{S} \;\; e \;\; a \in \mathcal{A}. \tag{16}
$$

De acordo com Bellman, uma política ótima é caracterizada de forma que qualquer que seja o estado inicial e a decisão inicial, as decisões subsequentes devem produzir uma otimalidade da política, relativamente ao estado resultante da primeira decisão.

[5] Richard Ernest Bellman foi um matemático norte-americano (1920 - 1984).

Com relação a uma política ótima, é possível afirmar que um valor de um determinado estado, nesse caso, deve ser igual ao valor de se tomar a ação que possui o maior valor estado-ação, podendo ser escrita como

$$v^*(s) \doteq \max q_{\pi*}(s, a). \tag{17}$$

No caso de $q_{\pi*}$, sua relação com o valor dos próximo estados, relativamente a uma política ótima, pode ser reescrita a partir de (12) como

$$q_{\pi*}(s, a) = \sum_{s',r} p(s', r|s, a)[r + \gamma v^*(s')]. \tag{18}$$

A partir das equações (17) e (18), pode-se chegar à função de estado ótimo, conhecida como a *equação da otimalidade de Bellman para o estado*, definida como

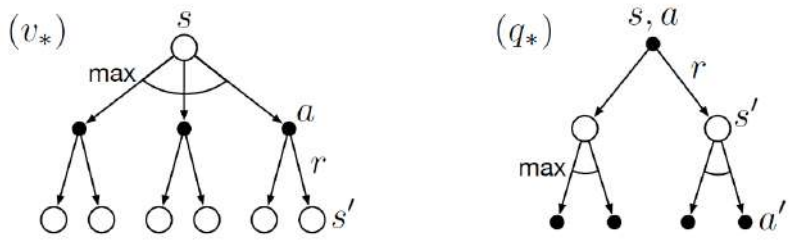
$$v^*(s) = \max \sum_{s',r} p(s', r|s, a)[r + \gamma v^*(s')]. \tag{19}$$

Uma relação semelhante pode ser obtida para o caso da função valor estado-ação, obtendo nesse caso a *equação de otimalidade de Bellman para o estado-ação*, escrita como

$$q^*(s) = \sum_{s',r} p(s', r|s, a)[r + \gamma \max q^*(s', a')]. \tag{20}$$

No entanto, essas equações de otimalidade de Bellman não precisam ser encaradas como um algoritimo, mas sim como um sistema de equações. Um diagrama dessas relações de otimalidade pode ser observado na Figura 5.

Figura 5 – Diagrama de otimalidade em RL.



Fonte: adaptado de (TEIXEIRA, 2016).

Em um problema de RL, caso a dinâmica do ambiente seja conhecida, então torna-se possível resolver os sistemas de equações da otimalidade Bellman utilizando os métodos da DP (SUTTON; BARTO, 2018). Contudo, os algoritmos clássicos de DP têm sua utilidade limitada

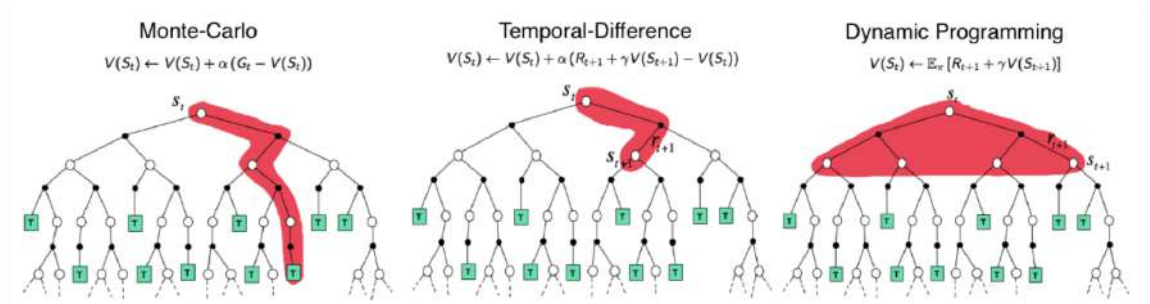
nesse caso, tanto por causa da suposição de um modelo perfeito do ambiente quanto por causa de sua grande despesa computacional, mas ainda são importantes teoricamente. De fato, os outros métodos de aprendizagem para estimar funções de valor e descobrir políticas ótimas, na realidade, podem ser encarados como tentativas de obter-se o mesmo efeito que a DP, apenas com menos computação e sem assumir um modelo perfeito do ambiente.

De forma oposta à DP, os algoritmos de Monte Carlo (MC) não necessitam ter um conhecimento completo do ambiente. Esses métodos, requerem apenas experiência, ou seja, amostras de sequências de estados, ações e recompensas da interação do agente com um ambiente, seja ela real ou em simulação. O aprendizado com a experiência real é um fato impressionante porque não requer um conhecimento prévio a respeito da dinâmica do ambiente, podendo atingir um comportamento ideal.

Os métodos de MC são formas de resolver o problema do RL com base na média dos retornos das amostras. De forma a garantir retornos bem definidos disponíveis, assume-se que os métodos de MC são aplicados apenas para tarefas episódicas. Em outras palavras, a experiência é dividida em episódios e que todos os episódios terminam eventualmente, independentemente das ações selecionadas. Somente após a conclusão de um episódio, as estimativas e políticas de valor são alteradas.

A união das ideias da DP e os métodos de MC dão origem a um tipo de aprendizagem muito útil na área do RL, a aprendizagem de diferença temporal (TD). Assim como nos métodos de MC, os métodos de TD podem aprender diretamente da "experiência bruta", sem a necessidade de um modelo perfeito da dinâmica do ambiente. Além disso, como na DP, os métodos TD atualizam as estimativas das funções baseados em parte de outras estimativas aprendidas, sem precisar esperar por um resultado final (são inicializados). Essas ideias e métodos se misturam e podem ser combinados de várias formas (Figura 6).

Figura 6 – Tipos de algoritmos para soluções de RL.



Fonte: adaptado de (LILLICRAP et al., 2016).

2.3.6 Q-learning

O Algoritmo de RL conhecido como *Q-Learning* proposto por (WATKINS, 1989), define uma função valor de estado-ação $Q(s, a)$, a qual representa a estimativa do valor ótimo. O valor ótimo é a maior recompensa futura, incluindo o desconto, a partir de um estado s, efetuando uma ação a e, mantendo-se a maior recompensa a partir desse ponto. Essa função é definida como

$$Q(s_t, a_t) = max(R_t). \tag{21}$$

A equação (21) representa uma medida de "qualidade" de um determinada ação a_t em um determinado estado s_t. De forma a realizar a estimativa do total de recompensas até um estado final, a partir do estado atual, sem conhecer quais serão os estados, ações e recompensas futuras, recorre-se à função de valor estado-ação $Q(s, a)$, escolhendo apenas as ações com maior valor a partir de um determinado estado,

$$\pi(s) = arg(max_a) = Q(s, a). \tag{22}$$

O processo de escolha de qual ação tomar durante o processo de aprendizado da função $Q(s, a)$ pode ser feito por meio de algum método de exploração ou até mesmo aleatoriamente (SILVA, 2015).

O cerne do algoritmo *Q-learning* é possibilidade da aproximação da função estado-ação utilizando a equação recursiva de Bellman. O pseudo-código para o *Q-learning* é descrito como:

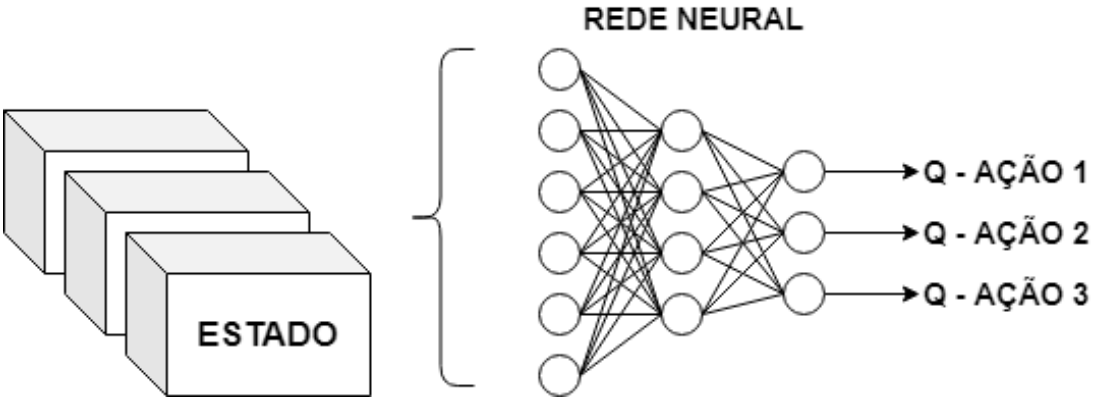
Algoritmo 1: Q-learning

1 Iniciar a função valor estado-ação aleatoriamente $Q(s, a)$;
2 Observar o estado inicial s_0 ;
3 **repeat**
4 Escolha uma ação a a partir de s utilizando a política;
5 Execute a ação a_t e observe a recompensa r_t e o próximo estado s_{t+1};
6 Atualize $Q(s, a) = Q(s, a) + \alpha_Q[r + \gamma max_{a'} Q(s', a') - Q(s, a)]$;
7 Faça $s \leftarrow s'$;
8 **until** s_t *é terminal*;

No algoritmo 1, os termos α_Q e γ representam as taxas de aprendizagem, controlando a relevância de quanto da diferença entre os valores, atual e o novo proposto para *Q*, será levado em consideração. Contudo, mesmo que teoricamente para atingir a convergência seja necessário que todos os valores estado-ação sejam muitas vezes acessados (WATKINS; DAYAN, 1992), na prática é visto que, executando-se um número elevado de iterações é possível atingir valores bastante significativos de convergência (SILVA, 2015).

A partir dos pares estado-ação, é possível implementar a função Q por meio de uma Tabela na qual as linhas são os estados e as colunas as ações, sendo denominada de representação tabular da função Q. Todavia, tal tratativa não é aconselhável para problemas mais complexos, os quais possuem um grande número de estados e/ou ações, tornando assim a representação tabular inviável. Uma forma de abordar tal situação é representar Q através de ANNs. Nesse contexto, as ANNs recebem um vetor de representação dos estados possíveis e são treinadas de modo a gerar saídas com valores de Q de cada ação possível, como ilustrado na Figura 7.

Figura 7 – Q-learning.



Fonte: próprio autor.

A atualização dos valores de Q se dá por meio da minimização da função perda/erro, ou função *loss*. Essa atualização utiliza o algoritmo de retro propagação do erro (*backpropagation*), método utilizado para calcular a contribuição de cada neurônio da rede no erro obtido.

2.3.7 Ator-Crítico

Na arquitetura conhecida como ator-crítico (AC), o agente de aprendizado faz uso de duas ANNs. A primeira tem como função avaliar o estado atual do ambiente (*crítico*) e a segunda, tem por objetivo definir a melhor ação para o respectivo estado atual (*ator*). A partir do erro da estimativa da rede de avaliação, obtêm-se as informações necessárias para atualização de ambas as redes. Em relação à estrutura do AC, as camadas ocultas e de entrada são iguais em ambas, no entanto, as camadas de saída são especificadas de acordo cada finalidade das redes.

A rede *crítico* representa a função valor estado $v(s)$, a qual estima a recompensa futura com desconto apenas a partir de um dado estado s, não necessitando da informação da ação específica, o que acaba por torná-la mais genérica, diferentemente da função de valor estado-ação $Q(s, a)$.

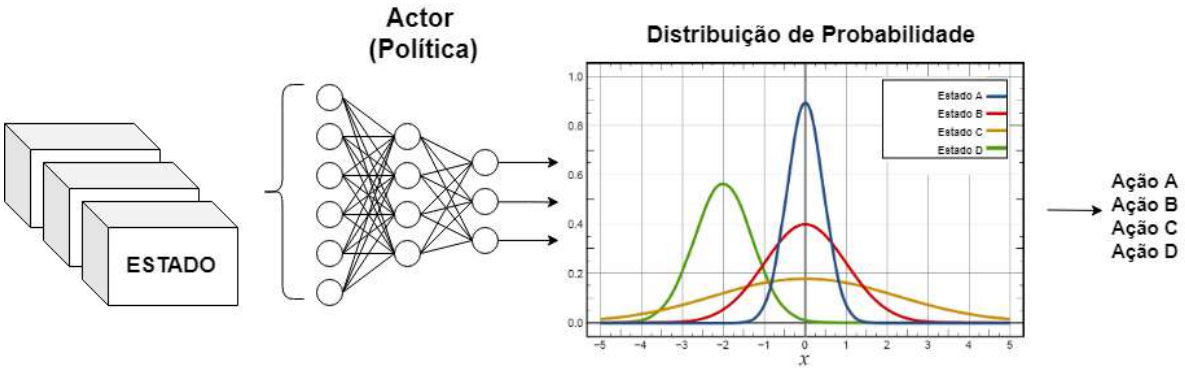
No *Q-Learning*, a estimativa do valor da recompensa futura considera, além do estado s, uma ação específica a, o que acaba por requerer sucessivas iterações de aproximação para

todos os valores possíveis de ações, isso torna o aprendizado, além de dispendioso para grandes quantidades de ações, inviável para ações no espaço contínuo.

No caso da rede *ator*, esta representa a política do agente, ou seja, a tomada de decisão para escolha da respectiva ação. De forma geral, é uma representação do mapeamento do estado s para os parâmetros de uma função categórica (discreta) ou mesmo gaussiana (contínua) de distribuição de probabilidade, a qual se dará o valor efetivo da ação, utilizando-se do valor médio e desvio padrão da distribuição.

A Figura 8 apresenta o modelo de ator crítico baseado em política de ações no modelo contínuo.

Figura 8 – Modelo de política com ação em modo contínuo da arquitetura AC.

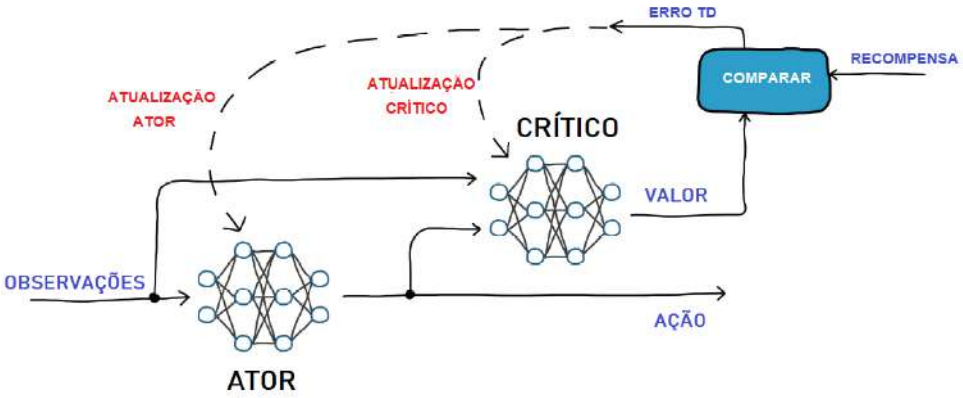


Fonte: próprio autor.

Com base nisso, fica clara a vantagem de aprender, além da política, a função valor estado, pois essa função contribui na atualização da política, reduzindo assim a variação do gradiente da função. De forma geral, a estrutura do AC (Figura 9) é resumida em:

- **Crítico** - Atualiza os parâmetros da função valor que, dependendo do algoritmo, pode ser a função valor estado-ação $Q(a|s)$ ou a função valor estado $V(s)$.
- **Ator** - Atualiza os parâmetros da política θ na direção sugerida pelo crítico, $\pi(a|s;\theta)$.

Figura 9 – Arquitetura AC.



Fonte: adaptado de (MATHWORKS, 2020).

Algoritmo 2: Ator-Crítico

Iniciar aleatoriamente s, θ, w;
Realize uma ação com os parâmetros inicializados $a \sim \pi(a, s; \theta)$
for *t=1, T* **do**
    Observe a recompensa $r_t \sim R(s, a)$ e o próximo estado $s' \sim P(s'|s, a)$;
    Selecione uma nova ação $a' = \pi_\theta(a'|s)$;
    Atualize os parâmetros da política: $\theta \leftarrow \theta + \alpha_\theta Q_w(s, a)\nabla_\theta ln\pi_\theta(a|s)$;
    Computar a correção (erro TD) para o valor-ação no tempo t:
    $\delta_t = r_t + \gamma Q_w(s', a') - Q_w(s, a)$
    use isto para atualizar os parâmetros da função valor-ação:
    $\omega = \omega + \alpha_t \delta_t \nabla_w Q_w(s, a)$;
    Atualize $a \leftarrow a'$ e $s \leftarrow s'$;
end

2.4 Aprendizado por reforço em aplicações de sistemas de controle

Os métodos de RL também podem ser aplicados diretamente aos problemas que envolvam os sistemas de controle dinâmicos. Quando um modelo do ambiente, nesse caso um sistema dinâmico, não está disponível, a estimação direta do valor da ação de controle, torna-se mais útil do que estimar o valor do estado do sistema. De forma geral, caso não exista o conhecimento do modelo do sistema, o valor do estado não é suficiente para determinar uma política de controle.

O comportamento de uma política no RL, observação do ambiente e geração de ações específicas para completar uma tarefa de maneira ideal, pode ser comparável a uma operação de um controlador em um sistema de controle dinâmico (Tabela 1). A analogia existente entre o RL e sistemas de controle é ilustrada na Figura 10.

Figura 10 – Associação entre sistemas de controle e RL.

Fonte: adaptado de (MATHWORKS, 2020).

Tabela 1 – Analogia entre a teoria do RL e sistemas de controle.

Aprendizado por Reforço	**Sistemas de Controle**
Ambiente	Todo o sistema ao qual o controlador está inserido, podendo incluir nesse caso: •Sinais de distúrbios • Filtros •Ruídos na medição •Conversores AD/DA
Observação	Qualquer valor mensurável do sistema dinâmico que o controlador é capaz de visualizar
Política	Controlador
Ação	Ações de controle ou variáveis manipuladas
Recompensa	Função de performance do sistema
Algoritmo de aprendizagem	Mecanismo de adaptação do controlador

O RL pode apresentar diversas aplicações práticas importantes em sistemas de controle dinâmicos. A partir da combinação do RL com controladores *feedback*, por exemplo, pode-se obter novas arquiteturas e melhores desempenhos dinâmicos para controladores tradicionais. Nesse sentido, o trabalho de (ANDERSON et al., 1997) realizou aplicações do RL em controladores *feedback* para realizar o controle do aquecimento e resfriamento em edifícios. Por outro lado, o trabalho de (GONÇALVEZ, 2016) fez uso de uma arquitetura AC para realizar

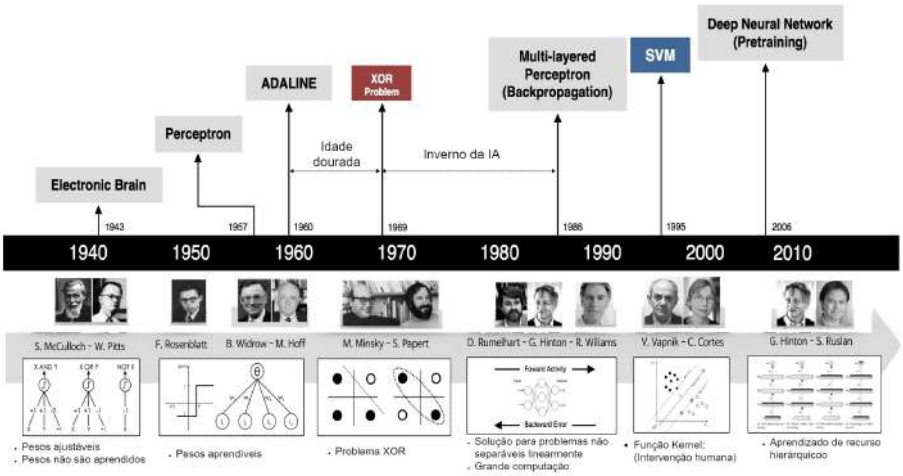
o controle adaptativo de um processo de nível. Além disso, observa-se que geralmente muitos problemas de controle possuem uma melhor tratativa por parte do RL ao se utilizar sinais de estados e de controle contínuos. Nesse sentido, o trabalho de (TU, 2001) apresentou uma metodologia utilizando RL em espaço contínuo, aplicando-a em um problema de controle simulado, o qual envolvia o refinamento de um controlador PI para o controle de uma planta simples.

No que diz respeito à teoria do controle robusto, o RL pode fornecer ferramentas para tratar situações práticas dos sistemas dinâmicos, os quais porventura possam apresentar características dinâmicas ruidosas, não lineares e até não especificadas para o controlador. Em (KRETCHMAR et al., 2001), utilizou-se um agente de RL em um modelo de sistema que apresentava características não lineares e incertezas em sua dinâmica. O resultado deste trabalho demonstrou que a estabilidade do sistema foi garantida, mesmo durante a aprendizagem do agente de RL.

2.5 Reforço profundo

A área do DL é uma subárea da ML, a qual utiliza determinados algoritmos para realizar um processamento massivo de dados. Nos últimos anos, o crescimento vertiginoso de trabalhos na área do DL proporcionou novas aplicações sobre os métodos tradicionais de ML, englobando uma variedade extensa de tarefas. Além disso, o DL representa uma grande evolução na utilização das ANNs (Figura 11), as quais possuem uma história que data à década de 1940.

Figura 11 – História das rede neurais artificiais.

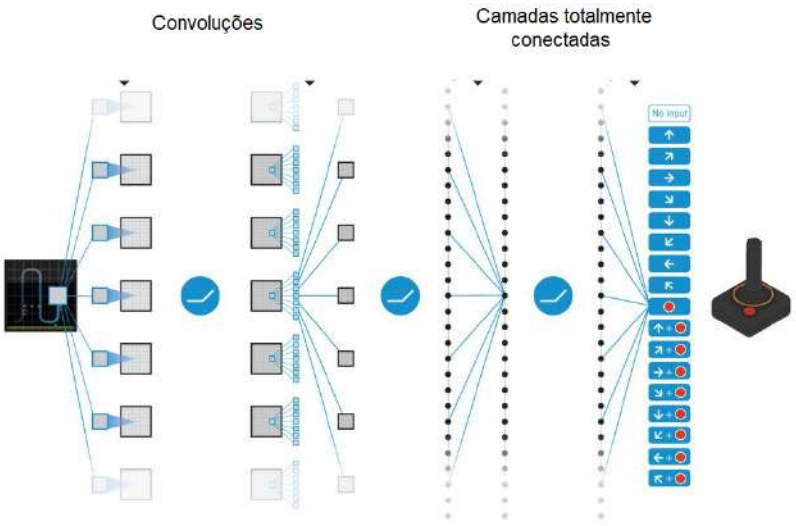


Fonte: adaptado de (ACADEMY, 2019).

Assim como no caso das ANNs, a DL marcou vários progressos na área do RL, formalizando uma área da ciência conhecida como o "aprendizado por reforço profundo" (DRL). Um dos marcos iniciais do DRL aconteceu a partir do desenvolvimento de um algoritmo

capaz de aprender a jogar os jogos de vídeo games do Atari 2600 (Figura 12) em um nível maior que o ser humano. A partir simplesmente dos *pixels*[6] de imagem da tela, fez-se possível atingir resultados que marcaram uma revolução na área do RL (MNIH et al., 2015). Este trabalho foi o primeiro a provar que os agentes de RL poderiam ser treinados a partir de observações brutas e com alta dimensionalidade, tomando como base apenas o sinal de recompensa, fornecendo desta maneira soluções para a instabilidade das técnicas de aproximação de funções em RL.

Figura 12 – Aprendizado por reforço profundo aplicado a jogos de Atari.



Fonte: adaptado de (MNIH et al., 2015).

Ainda a respeito deste tema, outro grande sucesso na área do DRL foi o desenvolvimento de um sistema híbrido, denominado *AlphaGo*, o qual derrotou o campeão mundial no jogo *Go*[7] (SILVER et al., 2016). Este sistema foi formulado a partir de ANNs treinadas utilizando um aprendizado supervisionado e também de reforço, combinando-se com um algoritmo de busca heurística tradicional.

De forma geral, algoritmos de DRL foram utilizados na abordagem de praticamente todos os tipos de tarefas de ML, passando pelo desenvolvimento de modelos de máquinas de última geração (ZOPH; LE, 2017) até a construção de novas funções de otimização (LI; MALIK, 2017). Nesse sentido, uma vez que o DRL tem sido utilizado em muitos ramos da ML, parece provável que, no futuro, o DRL seja um componente importante na construção de sistemas gerais de AI (ARULKUMARAN et al., 2017; LAKE et al., 2016).

[6] Elemento de imagem, menor ponto que forma uma imagem digital.

[7] Antigo jogo chinês estratégico de soma zero em que dois jogadores posicionam alternadamente pedras pretas e brancas em um tabuleiro.

2.5.1 Agentes do aprendizado por reforço profundo

De forma a melhor contextualizar o DRL, apresentam-se os agentes de DRL relevantes a título do presente trabalho.

2.5.1.1 DQN

Em RL os algoritmos apresentam instabilidade e até divergência quando uma função com características não lineares é utilizada para realizar a representação das ações, como no caso das ANNs (MATIISEN, 2017). De maneira a contornar tal situação no caso do algoritmo *Q-learning* que utiliza ANNs, o trabalho de (MNIH et al., 2013) propôs uma nova abordagem. Por meio da adição de técnicas que melhoraram sua estabilidade e, além disso, utilizando-se de DL para representar melhor a função valor (*Q*), foi possível obter um novo algoritmo capaz de sanar tais dificuldades, conhecido como *Deep Q-Network* (DQN). O pseudocódigo para o algoritmo DQN é descrito da seguinte forma:

Algoritmo 3: DQN

Iniciar a memória de reuso $\mathcal{D}$ com tamanho N;
Iniciar a função valor-ação *Q* com pesos aleatórios θ;
Iniciar a função valor-ação target $\hat{Q}$ com pesos $\theta^{-} = \theta$;
Observar o estado inicial s_0 ;
for *episódio=1, M* **do**
    Inicializar a sequência do episódio s_1;
    for *t=1, T* **do**
        Selecionar uma ação aleatória a_t com probabilidade ε
        Caso contrário selecionar a ação $a_t = argmax_a Q(s, a; \theta)$;
        Executar a ação a_t e observar a recompensa r_t e o próximo estado s_{t+1} ;
        Armazenar a transição (s_t, a_t, r_t, s_{t+1}) na memória $\mathcal{D}$;
        Selcione uma amostra aleatória do minibatch das transições $\mathcal{D}$;
        Faça $y_j = r_j$ se o episódio terminar em $t + 1$
        Do contrário faça $y_j = r_j + \gamma max_{a'} \hat{Q}(s_{t+1}, a'; \theta^{-})$;
        Compute o gradiente $(y_j - Q(s_t, a_t; \theta))^2$ com relação aos parâmetros θ da rede ;
        A cada C passos reset $\hat{Q} = Q$;
    end
end

Uma das formas do DQN melhorar a estabilidade no treinamento é através da chamada memória de reuso ou reutilização de experiência - *experience replay* ($\mathcal{D}$). Esta técnica cria uma memória das transições, ou experiências, as quais são utilizadas de forma aleatória durante a etapa de treinamento do agente. Desta maneira, evita-se *overfitting*, aumenta-se a velocidade de aprendizagem, além de reduzir-se a correlação entre as transições subsequentes, pois do

contrário, a similaridade poderia induzir a rede para um mínimo local. Além disso, utiliza-se uma rede de aproximação separada - *separate target network*, sendo basicamente um clone da rede principal. O objetivo disso é obter o valor de aproximação *Q-target* usado no cálculo da função perda/erro. Essa rede (clone da principal) é atualizada periodicamente a partir de uma determinada quantidade de transições e, caso apresente uma variação constante, seu treinamento será mais difícil.

2.5.1.2 DDPG

No trabalho de (LILLICRAP et al., 2016), intitulado "*continuous control with deep reinforcement learning*", adaptaram as ideias do DQN para espaço de ações contínuas, possibilitando realizar o controle de sistemas contínuos. Nesse trabalho foi desenvolvido o algoritmo denominado de Gradiente de Política Determinista Profunda - *Deep Deterministic Policy Gradient* (DDPG), o qual se utiliza de uma estrutura AC, enquanto aprende uma política determinística. O pseudocódigo para o algoritmo DDPG é formulado da seguinte forma:

Algoritmo 4: DDPG

Iniciar aleatoriamente as redes do crítico $Q(s, a|\theta^Q)$ e ator $\mu(s|\theta^\mu)$ com pesos θ^Q e θ^μ;
Inicializar as redes target Q' e μ' com pesos $\theta^{Q'} \longleftarrow \theta^Q$ e $\theta^{\mu'} \longleftarrow \theta^\mu$;
Inicializar o *replay buffer* R;
for *episódio=1, M* **do**
    Inicializar um processo aleatório N para a exploração das ações;
    Receber a observação inicial de estado s_1;
    for *t=1, T* **do**
        Selecionar uma ação $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}$ com base na política e ruído de exploração atuais;
        Executar a ação a_t e observar a recompensa r_t e o novo estado s_{t+1};
        Armazenar a transição (s_t, a_t, r_t, s_{t+1}) em R;
        Coletar uma amostra aleatória do minibatch de N transições (s_t, a_t, r_t, s_{t+1}) do *buffer*buffer R;
        Faça $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$;
        Atualize o crítico pela minimização da loss: $L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$;
        Atualize a política do ator usando a amostra do gradiente de política: $\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$
        Atualize as redes targets:
$$\theta^{Q'} \longleftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$$
$$\theta^{\mu'} \longleftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$
    end
end

Esse algoritmo utiliza quatro ANNs, *Q-network* (θ^Q), *deterministic policy function* (θ^μ),

target Q-network ($\theta^{Q'}$) e *target policy network* ($\theta^{\mu'}$). A *Q-network* e as redes de políticas mapeiam diretamente os estados para ações, em vez de gerar a distribuição de probabilidade em um espaço de ação discreto. Na realidade, as redes *target* são cópias com atraso de tempo de suas redes originais, as quais rastreiam lentamente as redes aprendidas. O uso dessas redes de valores *target* melhora muito a estabilidade no processo de aprendizagem. Nos métodos que não usam redes *target*, as equações de atualização da rede são interdependentes nos valores calculados pela própria rede, o que a torna propensa a divergências (LILLICRAP et al., 2016).

Nos algoritmos de RL que envolvem espaços de ação discretos, a exploração acontece por meio de uma seleção probabilística de uma ação aleatória (como *epsilon-greedy* ou *exploração de Boltzmann*). Por outro lado, neste caso a exploração é feita adicionando-se um ruído à própria ação. No trabalho original do DDPG (LILLICRAP et al., 2016), os autores utilizaram o processo *Ornstein-Uhlenbeck* para adicionar o ruído à saída da ação (UHLENBECK; ORNSTEIN, 1930).

2.5.1.3 TD3

O algoritmo conhecido como *Twin Delayed DDPG* (TD3) foi concebido a partir de (FUJIMOTO; HOOF; MEGER, 2018). O TD3 explora algumas deficiências do algoritmo DDPG que, apesar de ter um ótimo desempenho em diversos problemas, algumas vezes apresenta uma fragilidade no que diz respeito aos seus hiperparâmetros e alguns outros ajustes específicos. De forma geral, uma deficiência do DDPG é o caso deste superestimar a função Q o que acarreta na ruptura da política.

De forma a contornar algumas das deficiências do DDPG, o trabalho de (FUJIMOTO; HOOF; MEGER, 2018) propôs algumas alternativas, destacando-se:

- Aprendizado *clipped double-Q*. Nesse caso, aprendem-se duas *Q-functions* (’*twin*’)[8] e se utiliza do menor valor entre essas funções para formar os *targets* nas funções de erro de Bellman.
- Atualizações de políticas em atraso. A atualização da política e das redes *targets* com uma frequência menor do que a da própria *Q-function*
- Amenização da *target policy*. Um ruído é adicionado ao *target action*, tal que torne-se mais difícil a política explorar os erros da *Q-function* ao longo das mudanças na ação.

O TD3 realiza o treinamento em uma política determinística de maneira *off-policy*. Nesse sentido, uma vez que a política é determinística, caso o agente explorasse a política, provavelmente no início não haveria uma diversidade de ações suficientemente grande para

[8] Do inglês gêmeo.

encontrar sinais de aprendizagem eficazes. Por esta razão, o ruído[9] é adicionado às suas ações no momento do treinamento. O pseudocódigo do algoritmo TD3 é formulado da seguinte maneira:

Algoritmo 5: Twin Delayed DDPG

Input: parâmetros iniciais da política (θ), parâmetros da *Q-Function* (ϕ_1, ϕ_2), espaço da memória de reuso $\mathcal{D}$;
Definir parâmetros de destino iguais aos parâmetros principais $\theta_{targ} \leftarrow \theta$, $\phi_{targ,1} \leftarrow \phi_1$, $\phi_{targ,2} \leftarrow \phi_2$;
while *not convergece* **do**
    Observe o estado s e selecione a ação $a = clip(\mu_\theta(s) + \epsilon, a_{Low}, a_{High}), where\ \epsilon \sim \mathcal{N}$;
    Execute a no ambiente;
    Observe o novo estado s', nova recompensa r e o sinal *d* para indicar o término de s';
    Armazene (s, a, r, s', d) na memória de reuso *D*;
    if s' é o estado terminal, reset o ambiente;
    if *Tempo de atualização* **then**
        for *j no intervalo das atualizações* **do**
            Selecionar aleatoriamente um conjunto das transições B= (s, a, r, s') a partir de *D*;
            Compute *target action* $a'(s') = clip(\mu_{\theta_{targ}}(s') + clip(\epsilon, -c, c), a_{Low}, a_{High}), \quad \epsilon \sim \mathcal{N}(0, \sigma)$;
            Compute *targets* $y(r, s', d) = r + \gamma(1 - d) \min_{i=1,2} Q_{\phi_{targ,i}}(s', a'(s'))$;
            Atualize *Q-functions* através de um passo do gradiente descendete utilizando $\nabla_{\theta_i} \frac{1}{|B|} \sum_{(s,a,r,s',d) \in B} (Q_{\phi_i}(s, a) - y(r, s', d))^2 \quad for\ \ i = 1, 2$;
            if *j em modo policy-delay = 0* **then**
                Atualize a política através de um passo do gradiente ascendente utilizando $\nabla_\theta \frac{1}{|B|} \sum_{s \in B} (Q_{\phi_1}(s, \mu_\theta(s))$;
                Atualize as targets networks com $\phi_{targ,i} \leftarrow \rho \phi_{targ,i} + (1 - \rho)\phi_i \quad for\ i = 1, 2$ $\phi_{targ} \leftarrow \rho \phi_{targ} + (1 - \rho)\theta$;
            end
        end
    end
end

[9] Ruído gaussiano de média zero não correlacionado.

2.5.1.4 TRPO

O algoritmo conhecido como *Trust Region Policy Optimization* (TRPO) é fruto dos trabalhos (SCHULMAN et al., 2015; SCHULMAN et al., 2018). De forma geral, o TRPO realiza um atualização das políticas a partir do maior passo possível, tal que possa melhorar o desempenho e, ao mesmo tempo, satisfaz uma condição de restrição especial (*KL-Divergence*[10]), baseada em quão próximas as políticas antigas e novas podem estar (OPENAI, 2020). O pseudocódigo do TRPO é formulado da seguinte forma:

Algoritmo 6: Trust Region Policy Optimization

Input: parâmetros iniciais da política (θ_0), parâmetros iniciais da função valor (ϕ_0) ;
Definir hiperparâmetros: limite da divergência-KL δ, coeficiente *backtracking* α, número máximo de passos de backtrackings K;
for *k = 0, 1, 2, ...* **do**
    Coletar conjunto de trajetórias $\mathcal{D}_k = \{\tau_i\}$ através da política $\pi_k = \pi(\theta_k)$ no ambiente;
    Compute $\widehat{R}_t$;
    Compute as estimativas do $Advantage$ ($\widehat{A}_t$) - utilizando qualquer método para a estimação este termo - baseando-se na atual função valor V_{ϕ_k};
    Estime o gradiente da política como $\widehat{g}_k = \frac{1}{|\mathcal{D}|}\sum_{\tau\in\mathcal{D}_k}\sum_{t=0}^{T}\nabla_\theta log\pi_\theta(a_t|s_t)|_{\theta_k}\widehat{A}$;
    Use o algoritmo de gradiente conjugado para calcular $\widehat{x}_k \approx \widehat{H}_k^{-1}\widehat{g}_k$, onde $\widehat{H}_k$ é a Hessiana da média das amostras da divergência-KL;
    Atualize a política pelo *backtracking* com $\theta_{k+1} = \theta_k + \alpha^j\sqrt{\frac{2\delta}{\widehat{x}_k^T\widehat{H}_k\widehat{x}_k}}\widehat{x}_k$, onde $j = \{0, 1, 2, 3, ..., K\}$ é o menor valor que melhora a amostra da perda e satisfaz a amostra da restrição da divergência-KL;
    Ajustar a função valor através de regressão do erro médio quadrático: $\theta_{k+1} = arg\min_{\phi}\frac{1}{|\mathcal{D}_k|T}\sum_{\tau\in\mathcal{D}_k}\sum_{t=0}^{T}(V_\phi(s_t) - \widehat{R}_t)^2$, tipicamente realizado via um algoritmo gradiente descendente;
end

O TRPO pode ser usado para ambientes com espaços de ação discreta ou contínua. Este algoritmo é do tipo *on-policy*, ou seja, o algoritmo explora o problema a partir da amostragem de ações das versões mais recente de sua política estocástica. No geral, nota-se que ao passar do tempo de treinamento, a política torna-se menos aleatória. Este fato está relacionado com a regra de atualização que incentiva a exploração das recompensas que já foram encontradas.

[10] A divergência de Kullback-Leibler (entropia relativa) é uma medida de como uma distribuição de probabilidade é diferente de uma segunda distribuição de probabilidade de referência.

2.5.1.5 PPO

Assim como no caso do algoritmo TRPO, o algoritmo conhecido como *Proximal Policy Optimization* (PPO) busca dar o maior passo possível de melhoria em uma política utilizando os dados disponíveis. No entanto, diferentemente do TRPO que utiliza métodos mais complexos de segunda ordem para este propósito, o algortimo PPO faz uso de métodos de primeira ordem. Nesse sentido, o PPO é mais simples de implementar e aparenta ter um desempenho tão bom quanto o TRPO (OPENAI, 2020; SCHULMAN et al., 2017). O pseudocódigo do PPO é formulado de acordo com:

Algoritmo 7: Proximal Policy Optimization

Input: parâmetros iniciais da política (θ_0), parâmetros iniciais da função valor (ϕ_0) ;
for *k = 0, 1, 2, ...* **do**
  Coletar conjunto de trajetórias $\mathcal{D}_k = \{\tau_i\}$ através da política $\pi_k = \pi(\theta_k)$ no ambiente;
  Compute $\widehat{R}_t$;
  Compute as estimativas do $Advantage$ ($\widehat{A}_t$) - utilizando qualquer método para a estimação este termo - baseando-se na atual função valor V_{ϕ_k};
  Atualize a política maximizando o PPO-Clip objetivo:
  $\theta_{k+1} = arg\,\max_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t))\right)$
  Ajustar a função valor através de regressão do erro médio quadrático:
  $\theta_{k+1} = arg\,\min_{\phi} \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} (V_\phi(s_t) - \widehat{R}_t)^2$, tipicamente realizado via um algoritmo gradiente descendente;
end

O PPO realiza o treinamento de uma política estocástica, ou seja, explora o ambiente através de uma amostragem das ações com base na versão mais recente de sua política estocástica. Além disso, observa-se que ao longo do treinamento, a política torna-se menos aleatória, uma vez que a regra de atualização incentiva a explorar recompensas encontradas anteriormente.

2.5.1.6 SAC

O algoritmo conhecido como *Soft Actor-Critic* (SAC) busca otimizar uma política estocástica de forma *off-policy*, formando uma "ponte" entre a otimização de política estocástica e as abordagens com base no DDPG (HAARNOJA et al., 2018; HAARNOJA et al., 2019).

De forma geral, o SAC treina a política para maximizar um *trade-off*[11] entre a entropia[12] e o retorno esperado no ambiente. Nesse sentido, um aumento da entropia pode significar uma maior exploração de ações e, em muitos casos, acelerar o aprendizado e evitar ótimos locais

[11] O termo refere-se, geralmente, a perder uma qualidade ou aspecto de algo, ganhando em troca outra qualidade ou aspecto.
[12] Pode ser definida como uma medida de aleatoriedade na política.

ruins. O pseudocódigo do SAC é formulado da seguinte maneira:

Algoritmo 8: Soft Actor-Critic

Input: parâmetros iniciais da política (θ), parâmetros iniciais da *Q-function* (ϕ_1,ϕ_2), espaço da memória de reuso $\mathcal{D}$;
Definir parâmetros de destino iguais aos parâmetros principais $\phi_{targ,1} \leftarrow \phi_1$, $\phi_{targ,2} \leftarrow \phi_2$;
while *until not convergence* **do**
    Observe o estado s e selecione a ação $a \sim \pi_\theta(\cdot|s)$;
    Execute a no ambiente;
    Observe o novo estado s', nova recompensa r e o sinal *d* para indicar o término de s';
    Armazene (s, a, r, s', d) na memória de reuso $\mathcal{D}$;
    if s' é o estado terminal, reset o ambiente;
    if *Tempo de atualização* **then**
        for *j no intervalo das atualizações* **do**
            Selecionar aleatoriamente um conjunto das transições $\mathcal{B}$= (s, a, r, s', d) a partir de $\mathcal{D}$;
            Compute os *targets* para a *Q-functions*
            $$y(r, s', d) = r + \gamma(1 - d)\left(\min_{i=1,2} Q_{\phi_{targ,i}}(s', \tilde{a}') - \alpha\, log\pi_\theta(\tilde{a}'|s')\right);$$
            Atualize *Q-functions* através de um passo do gradiente descendete utilizando
            $$\nabla_{\theta_i} \frac{1}{|B|} \sum_{(s,a,r,s',d)\in B}(Q_{\phi_i}(s, a) - y(r, s', d))^2 \quad for\ \ i = 1, 2;$$
            Atualize a política através de um passo do gradiente ascendente utilizando
            $$\nabla_\theta \frac{1}{|B|} \sum_{s\in B}\left(\min_{i=1,2} Q_{\phi_i}(s, \tilde{a}_\theta(s)) - \alpha\, log\, \pi_\theta(\tilde{a}_\theta(s)|s)\right),$$ onde $\tilde{a}_\theta(s)$ é uma amostra de $\pi_\theta(\cdot|s)$ que é difenciável com respeito a θ;
            Atualize as targets networks com
            $$\phi_{targ,i} \leftarrow \rho\phi_{targ,i} + (1 - \rho)\phi_i \quad for\, i = 1, 2;$$
        end
    end
end

Assim como o TD3, o SAC tem o alvo compartilhado calculado a partir das *target Q-networks* e, utilizam-se do truque *clipped double-Q*. Por outro lado, diferencia-se do TD3 pelo fato das ações do próximo estado, utilizadas no *target*, serem provenientes da atual política e não de uma *target policy* (OPENAI, 2020).

2.5.1.7 A2C

Na arquitetura AC, o "crítico" estima a função valor, podendo ser referente à ação (*Q*) ou ao estado (*V*). O termo denominado *Advantage* é dado pela subtração de *Q* por *V*. Esta relação

indica o quanto é melhor realizar uma ação específica em comparação com a ação geral média em um determinado estado. Nesse caso, ao invés de estimar ANNs para ambos os valores de função, utilizam-se as equações de otimalidade de Bellman (19) e (20) para encontrar o termo denominado de *Advantage Actor-Critic*. A família do *Advantage Actor-Critic* tem duas variantes principais: o *Asynchronous Advantage Actor Critic* (A3C) e o *Advantage Actor Critic* (A2C) (MNIH et al., 2016). O pseudocódigo do algoritmo A2C é formulado da seguinte maneira:

Algoritmo 9: Advantage Actor-Critic

Assuma os vetores dos parâmetros θ e θ_v ;
Iniciar o contador do passo $t \leftarrow 1$;
Iniciar o contador do episódio $E \leftarrow 1$;
repeat
  Reiniciar os gradientes: $d\theta \leftarrow 0$ e $d\theta_v \leftarrow 0$;
  Fazer $t_{start} = t$;
  Coletar o estado s_t;
  repeat
    Selecionar a_t de acordo com a política $\pi(a_t|s_t;\theta)$;
    Coletar a recompensa r_t e observar o novo estado s_{t+1};
    $t \leftarrow t+1$;
  until *atingir o estado terminal* s_t *ou* $t - t_{start} == t_{max}$;
  $R = \begin{cases} 0 & para\ o\ estado\ terminal\ s_t \\ V(s_t,\theta_v) & para\ um\ estado\ não\ terminal\ s_t \end{cases}$;
  for $i \in \{t-1, ..., t_{start}\}$ **do**
    $R \leftarrow r_i + \gamma R$;
    Acumular os gradientes com respeito a θ :
    $d\theta \leftarrow d\theta + \frac{\partial \phi_{\theta'}(x_i)}{\partial \theta'} \left(g - max \left\{ 0, \frac{k_{g-\delta}^T}{||k||_2^2} \right\} k \right)$
    Acumular os gradientes com respeito a θ :
    $d\theta \leftarrow d\theta + \nabla_\theta log\, \pi(a_i|s_i;\theta)(R - V(s_i;\theta_v)) + \beta_e \partial H(\pi(a_i|s_i;\theta))/\partial\theta$
    Acumular os gradientes com respeito a θ_v:
    $d\theta_v \leftarrow d\theta_v + \beta_v(R - V(s_i;\theta_v))(\partial V(s_i;\theta_v)/\partial\theta_v)$;
  end
  Realizar a atualização de θ utilizando $d\theta$ e do θ_v utilizando $d\theta_v$;
  $E \leftarrow E+1$;
until $E > E_{max}$;

De forma geral, o A3C realiza um treinamento paralelo, utilizando de vários "trabalhadores" em ambientes paralelos que atualizam uma função de valor global (assíncrono). Por outro lado, o algoritmo A2C é uma variante de um único trabalhador do A3C. De forma empírica, verificou-se que o A2C produz um desempenho comparável ao A3C, embora apresente uma eficiência ligeiramente melhor (YOON, 2019).

2.5.1.8 ACER

O algoritmo *Sample Efficient Actor-Critic with Experience Replay* (ACER), proposto por (WANG et al., 2017b), reúne várias ideias de outros algoritmos de DRL. Nesse sentido, destacam-se o uso de vários trabalhadores, assim como o A3C, a implementação de um *buffer* de reprodução, como o DQN, a utilização de um método de otimização de política de região de confiança, semelhante ao utilizado no TRPO. De forma específica, o algoritmo introduz uma arquitetura AC com o *experience replay*. O pseudocódigo do ACER é dado da seguinte forma:

Algoritmo 10: ACER for continuos actions

Reiniciar os gradientes $d\theta \quad 0$ e $d\theta_{\upsilon} \quad 0$;

Inicializar os parâmetros $\theta' \quad \theta$ e $\theta_{\upsilon} \quad 0$;

Amostra da trajetória $\{x_0, a_0, r_0, \mu(\cdot|x_0), ..., x_k, a_k, r_k, \mu(\cdot|x_k)\}$ a partir da memória de reuso $\mathcal{D}$;

for $i \in \{0, 1, 2, 3, ..., k\}$ **do**

  Calcular $f(\cdot|\phi_{\theta'}(x_i)), V_{\theta'_{\upsilon}}(x_i), \tilde{Q}_{\theta'_{\upsilon}}(x_i, a_i)$ e $f(\cdot|\phi_{\theta_a}(x_i))$;

  Amostra de $a'_i \sim f(\cdot|\phi'_{\theta}(x_i))$;

  $\rho_i \leftarrow \frac{f(a_i|\phi_{\theta'}(x_i))}{\mu(a_i|x_i)}$ e $\rho'_i \leftarrow \frac{f(a'_i|\phi_{\theta'}(x_i))}{\mu(a'_i|x_i)}$;

  $c_i \leftarrow min\{1, (\rho_i)^{\frac{1}{d}}$;

end

$Q^{ret} \quad \begin{cases} 0 & para\ o\ término\ de\ x_k \\ V_{\theta'_{\upsilon}}(x_k) & caso\ \ contrário \end{cases}$;

$Q^{opc} \quad Q^{ret}$;

for $i \in \{k-1, ..., 0\}$ **do**

  $Q^{ret} \leftarrow r_i + \gamma Q^{ret}$

  $Q^{opc} \leftarrow r_i + \gamma Q^{opc}$;

  Calcular as quantidades necessárias para a atualização da região de confiança

  $g \quad min\{c, \rho_i\}\nabla_{\phi_{\theta'}(x_i)} log\, f(a_i|\phi_{\theta'}(x_i))\,(Q^{opc}(x_i, a_i) - V_{\theta'_{\upsilon}}(x_i)) + \left[1 - \frac{c}{\rho'_i}\right]_+ (\tilde{Q}_{\theta'_{\upsilon}}(x_i, a'_i) - V_{\theta'_{\upsilon}}(x_i))\, log\, f(a'_i|\phi_{\theta'}(x_i))$

  $k \leftarrow \nabla_{\phi_{\theta'}(x_i)} D_{KL}[\, f(\cdot|\phi_a(x_i)||f(\cdot|\phi'_{\theta}(x_i)]$;

  Acumular os gradientes com respeito a θ :

  $d\theta \quad d\theta + \frac{\partial \phi_{\theta'}(x_i)}{\partial \theta'}\left(g - max\left\{0, \frac{k_{g-\delta}^T}{||k||_2^2}\right\} k\right)$

  Acumular os gradientes com respeito a θ'_{υ} :

  $d\theta_{\upsilon} \quad d\theta_{\upsilon} + (Q^{ret} - \tilde{Q}_{\theta'_{\upsilon}}(x_i, a_i))\nabla_{\theta'_{\upsilon}} \tilde{Q}_{\theta'_{\upsilon}}(x_i, a_i)$

  $\theta'_{\upsilon} : d\theta_{\upsilon} \quad d\theta_{\upsilon} + min\{1, \rho_i\}(Q^{ret}(x_t, a_i) - \tilde{Q}_{\theta'_{\upsilon}}(x_t, a_i))\nabla_{\theta'_{\upsilon}} V_{\theta'_{\upsilon}}(x_i)$;

  Atualize *Retrace target*: $Q^{ret} \quad c_i\left(Q^{ret} - \tilde{Q}_{\theta'_{\upsilon}}(x_i, a_i)\right) + V_{\theta'_{\upsilon}}(x_i)$

  Atualize *Retrace target*: $Q^{opc} \quad c_i\left(Q^{opc} - \tilde{Q}_{\theta'_{\upsilon}}(x_i, a_i)\right) + V_{\theta'_{\upsilon}}(x_i)$;

end

2.5.1.9 ACKTR

O *Actor Critic using Kronecker-Factored Trust Region* (ACKTR) é um algoritmo do tipo AC, proposto por (WU et al., 2017a). O algoritmo ACKTR combina a arquitetura AC à otimização da região de confiança para realizar uma melhoria na convergência do algoritmo e, além disso, faz uso da chamada fatoração de Kronecker[13] com o objetivo de melhorar a eficiência e escalabilidade das ações.

Em seu trabalho, (WU et al., 2017a) propôs o uso de curvatura de aproximação fatorada de Kronecker (*K-FAC*)[14] para fazer a atualização do gradiente para as redes do crítico e do ator. Nesse sentido, *K-FAC* proporciona uma melhoria no cálculo do gradiente natural[15] , o qual difere bastante do gradiente padrão.

2.6 Considerações finais

Uma das características mais relevantes na área do RL moderno é a interação significativa com outras disciplinas científicas. O RL pode ser considerado como sendo parte de uma tendência de anos da inteligência artificial, objetivando uma maior integração com as áreas da estatística, otimização, engenharia de controle, entre outros (BRADTKE; BARTO, 1996; BRADTKE; YDSTIE; BARTO, 1994). Ainda neste contexto, o RL realizou interações com as áreas da psicologia e neurociência, o que acabou por proporcionar benefícios substanciais em ambos os lados (DOLAN; DAYAN, 2013; FIORILLO; YUN; SONG, 2013; ARULKUMARAN et al., 2017).

O recente desenvolvimento do DRL permitiu aos pesquisadores obter melhores desempenhos em diversos problemas, o que porventura estimulou o crescente interesse nesta área da ciência (MNIH et al., 2015; LILLICRAP et al., 2016; MATIISEN, 2017). Além disso, os diversos algoritmos de DRL têm permitido, cada vez mais, o desenvolvimento de sistemas com maior precisão e inspiração em sistemas de aprendizado biológico (SCHRITTWIESER et al., 2020; AKKAYA et al., 2019; HAFNER et al., 2020).

---

[13] Leopold Kronecker foi matemático alemão (1823 - 1891).

[14] Método de otimização de segunda ordem para aprendizado que aproxima a curvatura por fatoração de Kronecker e reduz a complexidade de computação das atualizações de parâmetros.

[15] Do inglês Natural Gradient Descent ou Gradiente Descente Natural (NGD) é um método de otimização proposto por Shun-Ichi Amari em 1998 baseado na Geometria da Informação.

3 CONTROLE EMOCIONAL

A ciência cognitiva é um ramo que tem despertado bastante interesse ao longo dos tempos, sendo tema de diversos trabalhos fascinantes (SUN; BOOKMAN, 1994).

3.1 Introdução

O campo da ciência cognitiva é bastante interdisciplinar. Por meio da reunião de modelos computacionais da inteligência artificial e de técnicas experimentais de diversas áreas como a psicologia, por exemplo, busca-se desenvolver o entendimento mais preciso a respeito dos processos de funcionamento da mente humana.

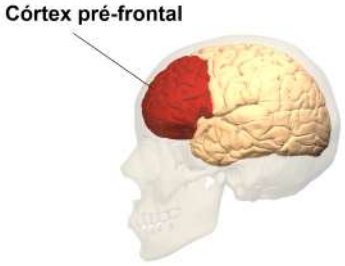
3.2 O sistema de recompensa do cérebro

Nos cérebros de muitos animais, existe um sistema, evolutivamente muito antigo, que processa as informações relacionadas aos sentimentos de prazer ou satisfação, conhecido como sistema (circuito) de recompensa cerebral.

O sentimento de prazer é uma característica motivacional, como explica o psiquiatra Rodrigo Grassi (FRANCKE; FERREIRA; GRASSI, 2010), "*Ele existe pela sobrevivência. O animal precisa ter algo que o motive a buscar alimento ou sexo, por exemplo. São elementos que são caros para a sobrevivência dele ou da espécie. É um sistema arcaico"*.

Todavia, grande parte dos prazeres da vida contemporânea não são essenciais ou importantes para a sobrevivência e, em muitos casos, podem trazer danos à saúde. Nesse sentido, faz-se necessário utilizar uma região relativamente mais jovem do cérebro, o córtex pré-frontal (Figura 13). A função do córtex pré-frontal é a da racionalidade mediante um sentimento de prazer. Essa região faz parte do sistema de recompensa, fornecendo a capacidade da ponderação sobre um sentimento, avaliando seus efeitos a longo prazo.

Figura 13 – Córtex pré-frontal.

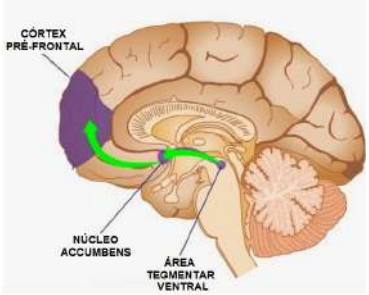


Fonte: adaptado de (HILL, 2017).

Nas situações onde ocorram experiências que forneçam uma sensação de prazer é

produzida uma substância conhecida como *dopamina*, um neurotransmissor que ativa o sistema de recompensa. A dopamina está localizada na parte ventral, conhecida como região rudimentar do cérebro, o que garante uma força biológica. A dopamina circula por algumas outras regiões do cérebro, chegando até o córtex pré-frontal, onde é feita a avaliação de satisfação. A Figura 14 ilustra o circuito no cérebro responsável pela recompensa.

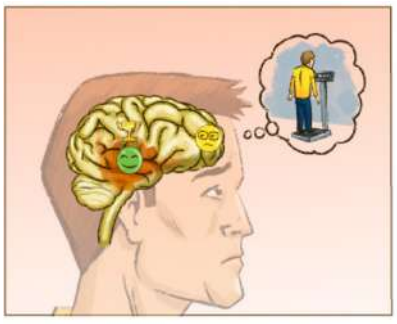
Figura 14 – Circuito de recompensa no cérebro.



Fonte: adaptado de (ACHUTTID, 2010).

No caso de um evento provocar uma sensação de felicidade, como por exemplo, comer um chocolate, o cérebro faz um registro de que isso é prazeroso e, portanto, tal situação precisa ser repetida. A partir do momento em que o cérebro associa comer chocolate a uma sensação prazerosa, a área tegmental ventral solicita a repetição da ação, ou seja, cria uma força biológica para isso. No entanto, quando essa informação chega no córtex pré-frontal, atua o processo de racionalidade, o que dá início a uma "batalha" entre a razão (córtex pré-frontal) e a emoção (tegmental ventral), como ilustrado na Figura 15. Caso as decisões sejam frequentemente tomadas por conta da emoção em relação a um comportamento prazeroso, consequentemente será mais difícil resistir a tal situação.

Figura 15 – Etapa de decisão entre a escolha racional ou emocional.



Fonte: adaptado de (FRANCKE; FERREIRA; GRASSI, 2010).

No caso das decisões não serem frequentemente baseadas na emoção, o cérebro não irá deixar de associar a ação de comer um chocolate com o prazer, por exemplo, mas pode encontrar

o prazer também em um alimento diferente e saudável, como uma fruta. Nesse caso, comer uma fruta frequentemente e passar a sentir-se bem com isso, pode ativar o sistema de recompensa e criar um ciclo específico. No entanto, a insistência em um determinado comportamento não produz interesse, por si só, nem todo comportamento traz o mesmo efeito para as diferentes pessoas. Deve-se portanto, encontrar um equilíbrio entre o chocolate e a fruta, ou seja, um equilíbrio vital entre a racionalidade e emoção, função primordial do sistema de recompensa cerebral.

3.3 Sistema de controle baseado no aprendizado emocional do cérebro

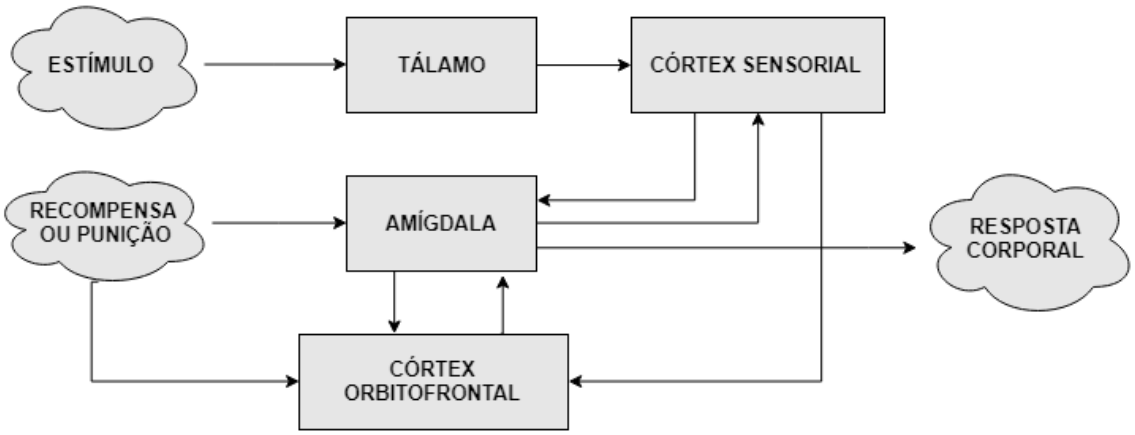
As características dos sistemas de aprendizado biológico, principalmente baseadas no cérebro humano, motivaram o desenvolvimento e aplicação de novas técnicas de aprendizado em controle adaptativo, resultando em novos modelos para aplicações na engenharia de controle moderno. Esses sistemas podem adaptar-se a novas condições ambientais, tais como incertezas nas medições sensoriais, variações paramétricas de sistemas dinâmicos e, a partir disso, contornar tais situações indesejadas por meio de uma resposta rápida, melhorando assim o desempenho do controle como um todo.

A partir de estudos e modelos relacionados a este tema, (MORéN; BALKENIUS, 2000; LUCAS; RASHIDI; ABDI, 2004), (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004) propuseram uma nova arquitetura de controlador, um controle com base no aprendizado emocional denominado de controlador inteligente baseado no aprendizado emocional do cérebro (BELBIC). O BELBIC é um controlador bio-inspirado pertencente à classe de controles *model-free*[1], tendo aplicações bem sucedidas, como por exemplo, na incorporação de controladores PID (ROUHANI et al., 2007), controlador de veículo de lançamento aeroespacial (MEHRABIAN; LUCAS; ROSHANIAN, 2006), robô omnidirecional (SHARBAFI; LUCAS; DANESHVAR, 2010), controle de carga-frequência (FARHANGI; BOROUSHAKI; HOSSEINI, 2012) e muitas outras aplicações na área de engenharia de controle (RAHMAN et al., 2008; MARKADEH et al., 2011).

O modelo de aprendizagem deste tipo de controlador é denominado de aprendizado emocional do cérebro - *Brain Emotional Learning* (BEL), o qual é inspirado no mecanismo de funcionamento do sistema límbico cerebral dos mamíferos. Este módulo consiste na aproximação de algumas regiões do cérebro como a amígdala, córtex orbitofrontal, entre outras. O modelo do BEL (Figura 16) deu origem a um modelo de sistema inteligente capaz de um aprendizado rápido no que tange à escolha de decisões das ações de controle, o que proporcionou grande aplicabilidade em diversas áreas de sistemas de controle (MORéN, 2002).

[1] Do inglês model-free ou livre de modelo, ou seja, nenhum modelo do processo está disponível e, portanto, o controlador não é projetado para um sistema específico.

Figura 16 – Estrutura do modelo emocional do cérebro baseada na amígdala.



Fonte: Adaptado (BEHESHTI; HASHIM, 2010).

3.3.1 Mecanismo emocional

Uma parcela significativa da literatura concorda que a emoção pode ser definida como sendo uma tendência de ação, possivelmente acompanhada por uma resposta psicológica. Estudos a respeito deste tema datam do século XIX, época em que desenvolveu-se um grande interesse a respeito dos processos cognitivos. O estudo das emoções engloba diversas áreas como a biologia, psicanálise e psicologia, o que contribui para a avaliação da relação que os efeitos de sentimentos emocionais exercem em diversas áreas da vida, seja em aspectos comportamentais, filosóficos ou de cognição (ANTONIO et al., 2008).

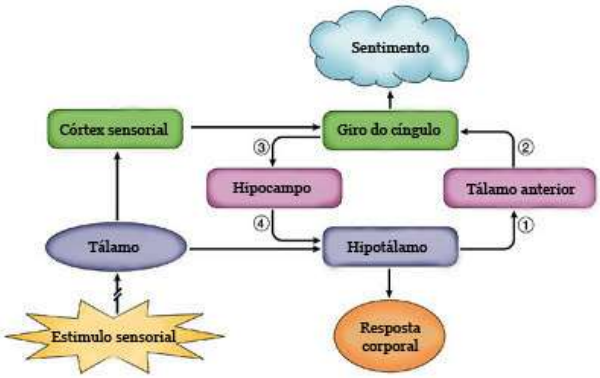
A teoria de James[2]-Lange[3] (CANNON, 1927) proposta no final do século XIX, afirma que as emoções nada são além do que a simples experiência do conjunto de alterações corporais que ocorrem em resposta a estímulos emocionais. No início do século XX, o trabalho de (CANNON, 1931; BARD, 1928) discordou da teoria de James-Lange, pois verificou-se em experimentos com gatos decorticados[4], a presença de ataques súbitos de raiva, denominado de "falsa raiva". Caso as emoções fossem um resultado direto da percepção de mudanças corporais, elas então seriam totalmente dependentes da integridade do córtex sensorial e motor. A partir disso, esse trabalho propôs que o hipotálamo seria a principal região do cérebro envolvida nas respostas emocionais e que tais respostas seriam inibidas por regiões corticais mais recentes, relativamente ao processo evolucionário. Nesse caso, a retirada das regiões corticais "libera" o circuito hipotalâmico, permitindo que a emoção não controlada se manifeste, assim como "falsa raiva".

O norte-americano Joseph Papez em 1937 (PAPEZ, 1937), propôs o famoso *circuito de Papez* que esquematiza o circuito neural central das emoções (Figura 17). Nesse sentido, Papez

[2] William James foi filósofo e psicólogo norte-americano (1842 - 1910).
[3] Carl Georg Lange foi um médico e psicólogo dinamarquês (1834 - 1900).
[4] Animais que têm partes cerebrais completamente removidas.

afirmou que as mensagens sensoriais relacionadas aos estímulos emocionais que chegam ao tálamo são dirigidas tanto para o córtex (corrente de pensamento) quanto para o hipotálamo (corrente de sentimento).

Figura 17 – Circuito de Papez.



Fonte: adaptado de (PAPEZ, 1937).

A teoria do aprendizado emocional baseia-se na avaliação emocional, em outras palavras, a percepção dos estímulos externos e suas possíveis consequências sobre o funcionamento do sistema em um curto prazo, objetivando manter a sobrevivência a longo prazo. No entanto, além dos sinais externos, provenientes do meio ambiente, os estados internos, emocional e cognitivo, exercem uma fundamental importância no aprendizado, muitas vezes de maior relevância do que aqueles estímulos externos. O sistema límbico é o circuito neural que controla o comportamento emocional e as forças emocionais, ou seja, o processamento das emoções. Este sistema está relacionado com a natureza afetiva das emoções, relacionado-as à recompensas ou punições. Na ciência cognitiva, as recompensas são modeladas como sinais positivos, enquanto as punições como sinais negativos. O sistema de recompensa e punição sem dúvida se constitui nos controladores mais importantes nas áreas das atividades físicas, além de controlar os desejos, aversões e motivações (LEDOUX; FELLOUS, 1995; PANKSEPP, 1981).

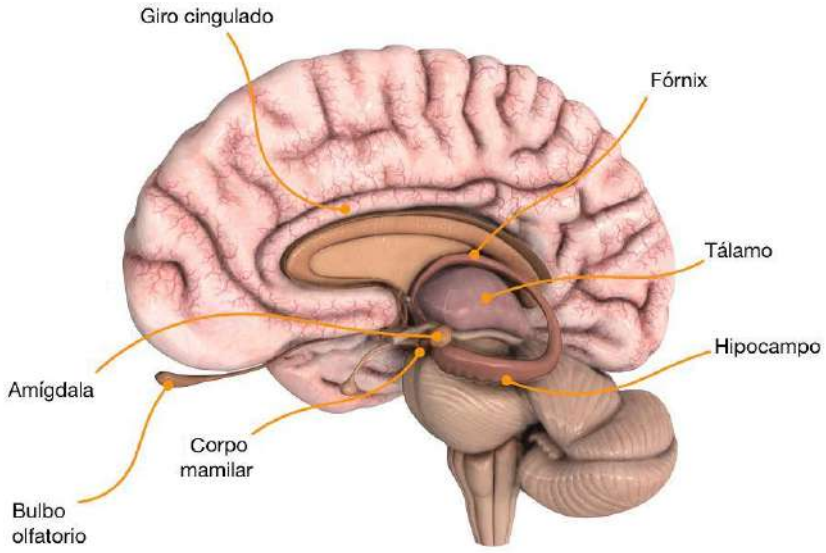
De maneira geral, o papel das emoções é centrado na avaliação dos estímulos recebidos e suas respectivas contribuições, de forma que o sistema alcance um determinado objetivo de maneira menos dispendiosa. Assim sendo, pode-se afirmar que o controle emocional visa economizar recursos que, em vez de avaliar todos os possíveis estímulos sensoriais, a avaliação concentra-se nos sinais que realmente são relevantes, ou seja, aqueles que têm o potencial de ser mais decisivos no processo de controle.

3.3.2 Sistema Límbico

Os estudos de (AFIFI; BERGMAN, 1997) abordaram a importância das estruturas do lobo temporal concernentes às emoções, sendo fundamental para a teoria de MacLean[5]. A principal ideia de MacLean (MACLEAN, 1952) era a de que as experiências emocionais envolvem a integração das sensações captadas do mundo externo com a informação corporal. Sua proposta foi de que eventos externos levam a mudanças no corpo. As mensagens a respeito de tais modificações retornam para o cérebro, onde são integradas com a percepção do mundo externo, gerando a experiência emocional. Em seu trabalho (MACLEAN, 1949) postulou que essa integração era realizada no hipocampo e, anos depois, introduziu-se o termo *sistema límbico*. Além disso, esse sistema é responsável por algumas funções vitais no corpo humano como por exemplo o controle da temperatura corporal, motivação e um grande impacto no processo de formação da memória (GUYTON; HALL, 2006). Nos seres humanos (Figura 18), o sistema límbico tem um maior desenvolvimento, o que acaba por proporcionar uma maior flexibilidade em relação às mudanças do meio ambiente.

A anatomia e o funcionamento do sistema límbico são bastante complexos, o que acarreta em grande desafio por parte dos pesquisadores com relação ao seu estudo e entendimento. No entanto, sabe-se que dentre as diversas estruturas que formam o sistema límbico as principais são a amígdala, hipocampo, córtex pré-frontal e o hipotálamo. Outras estruturas como o núcleo accumbens, septo, ínsula, córtex somatossensorial e tronco cerebral também estão implícitos no sistema límbico, porém de menor relevância (SARANT; NETSKY, 1974).

Figura 18 – Sistema límbico.



Fonte: adaptado de (3D4MEDICAL, 2019).

[5] Paul MacLean foi um médico e neurocientista norte-americano (1913 - 2007).

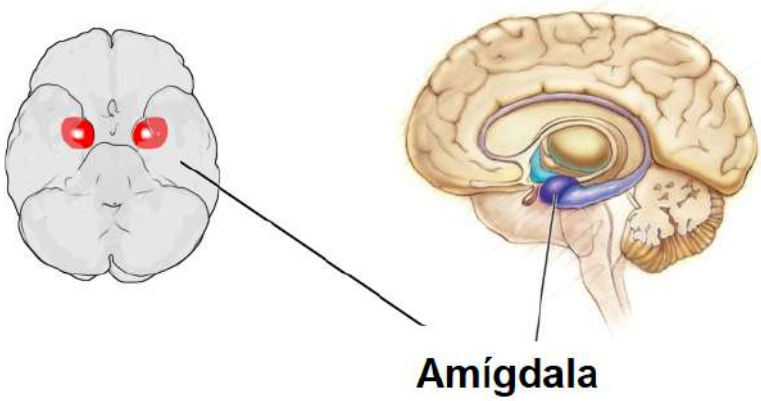
A amígdala e o hipocampo são considerados duas regiõpes primordiais do sistema límbico. A região do hipocampo é considerada como uma área responsável pela chamada memória de curto prazo. No caso da amígdala, sua função está diretamente relacionada com as emoções como por exemplo a alegria e a raiva (SARANT; NETSKY, 1974).

3.3.2.1 Amígdala

A amígdala é uma das mais importantes estruturas cerebrais relativas às emoções. Experimentalmente, através de sinais elétricos, observou-se que sua estimulação provoca sentimentos de agressão, medo e ansiedade. A sensação de medo, por exemplo, é fundamental para a sobrevivência individual e o autocontrole, promovendo uma avaliação emocional instantânea de uma situação ainda mais eficiente do que o raciocínio e a lógica complexos. Em suma, a amígdala é o centro integrador de emoções, comportamento emocional e motivação (PURVES et al., 2011).

A localização da amígdala se dá em uma região conhecida como subcortical, seu formato é semelhante a uma amêndoa (Figura 19), tendo comunicação direta com todos os outros córtices sensoriais e áreas dentro do sistema límbico.

Figura 19 – Amígdala.



Fonte: adaptado de (BELRESEARCH, 2017).

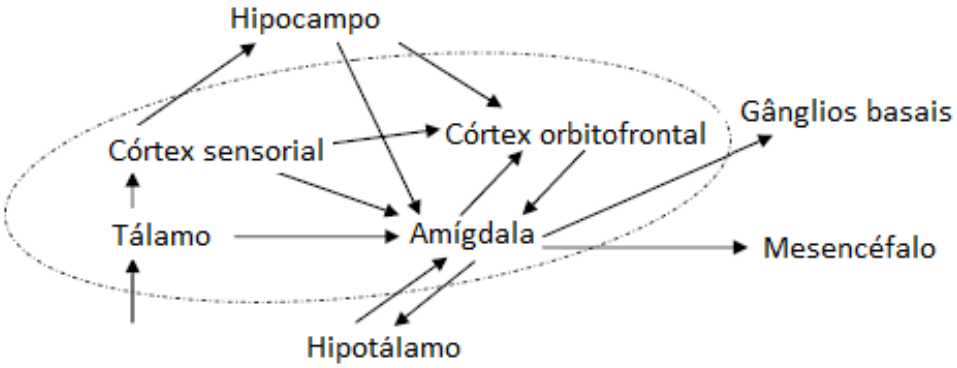
Durante a etapa de aprendizagem emocional do cérebro, a amígdala participa na reação aos estímulos emocionais, armazenando respostas emocionais (HOOKER et al., 2006), avaliando ambos os reforços positivos e negativos (FERRY B. ANDROOZENDAAL; MCGAUGH, 1999), aprendendo a associação entre estímulos incondicionados e aqueles condicionados (KANDEL; SCHWARTZ; JESSELL, 2003). Além disso, prevê a associação entre estímulos e reforços

futuros (ARMONY; LEDOUX, 1997) e a formação de uma associação entre estímulos neutros e estímulos emocionalmente carregados (KANDEL; SCHWARTZ; JESSELL, 2003).

As duas partes principais da amígdala são a parte basolateral (a maior porção da amígdala) e a parte centro-medial. A parte basolateral tem ligação bidirecional ao córtex insular e ao córtex orbital (ARMONY; LEDOUX, 1997) e desempenha o papel principal na mediação da consolidação da memória (ROBERTS, 2006), fornecendo a resposta primária, sendo dividida em três partes: basal lateral, basal e acessória (HOOKER et al., 2006). O basal lateral é a parte através da qual os estímulos entram na amígdala. A região lateral não apenas passa os estímulos para outras regiões, mas também os memoriza para formar a associação de resposta ao estímulo (KANDEL; SCHWARTZ; JESSELL, 2003). Essa parte também desempenha alguns papéis na divulgação das informações do sensor para outras partes, formando associação entre os estímulos condicionados e incondicionados, memorizando as experiências emocionais. As partes basais e acessórias basais participam na mediação do condicionamento contextual (DALGLEISH, 2004).

A amígdala tem conexão com o hipocampo, tálamo, núcleos septais e com o córtex pré-frontal. A partir disso, garante-se um importante desempenho na mediação e controle das atividades emocionais. A Figura 20 apresenta um esquema das conexões da amígdala com outros componentes do sistema límbico.

Figura 20 – Estrutura do modelo emocional do cérebro baseada na amígdala.

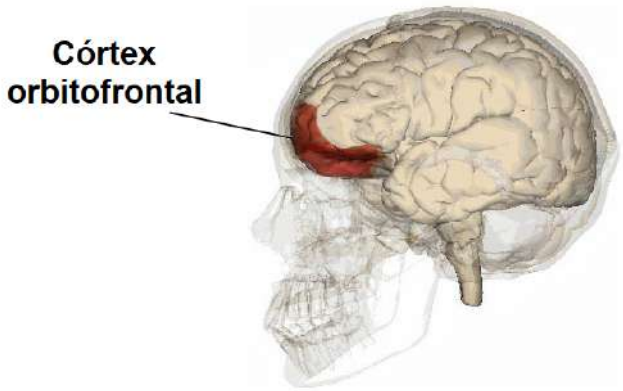


Fonte: adaptado de (BEHESHTI; HASHIM, 2010).

3.3.2.2 Córtex Orbitofrontal

Enquanto a amígdala aprende associações entre os estímulos emociaonais, o córtex orbitofrontal (Figura 21) inibe a expressão dessas associações, conforme necessário, dependendo do contexto e de outros fatores.

Figura 21 – Córtex orbitofrontal.



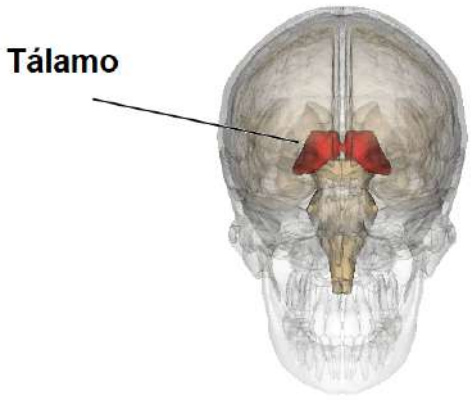
Fonte: adaptado de (ATLAS, 2017).

Além disso, a região orbitofrontal desempenha várias funções importantes no sistema límbico, entre as quais estão a capacidade de gerar sinais negativos de reforço. Esses sinais visam mitigar quaisquer respostas que porventura sejam inapropriadamente geradas pela amígdala. O córtex orbitofrontal opera com base na diferença entre recompensa ou punição esperada e as reais. A motivação esperada fica impressa nas estruturas do cérebro ao longo do tempo como resultado de vários mecanismos de aprendizagem e atinge o córtex orbitofrontal através do córtex sensitivo e da amígdala. A recompensa ou punição real vem do mundo exterior. Se os sinais de motivação esperados e detectados forem idênticos, a saída é a resposta regular a esse estímulo. Caso contrário, o córtex orbitofrontal suprime a resposta emocional típica e promove maneiras de aprender mais. Esse aprendizado e adaptação é um elemento-chave para a robustez dos organismos quando submetidos a ambientes em constante mudança que, no caso de sistemas dinâmicos, associa-se às pertubações e até mudanças na dinâmica do sistema. Ao mesmo tempo, a robustez às mudanças de condições também é uma característica fundamental de qualquer problema de engenharia. Assim, nos últimos anos, o comportamento emocional do cérebro tem sido objeto de atenção como um novo paradigma no campo do design de sistemas de controle. Vale salientar que o modelo abordado de sistema límbico, pode diferir em algumas literaturas (FUSTER, 2006; SHIMAMURA, 1995; KOLB; WHISHAW, 2009).

3.3.2.3 Tálamo

O tálamo é baseado na região subcortical, precisamente ao lado dos gânglios basais (Figura 22). Apesar de ser uma estrutura homogênea, é composta por uma série de áreas menores, aparentando comportamento independente.

Figura 22 – Tálamo.



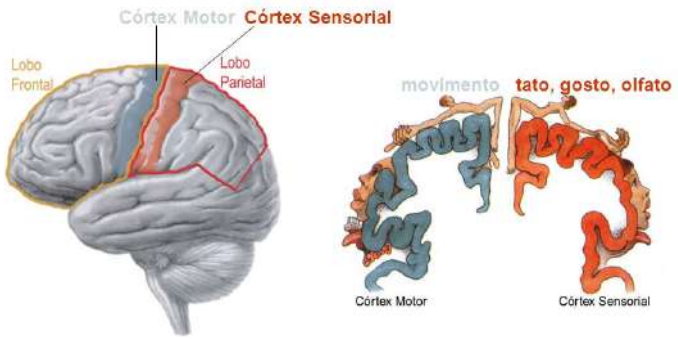
Fonte: adaptado de (MEDICALSCHOOL, 2018).

O tálamo se comporta como uma estação de comutação sensorial que reúne e pré-processa os dados sensoriais. Com exceção do sentido olfativo, todas as outras sensações são guiadas pela rede de nervos em direção ao tálamo. Por sua vez, encaminha essas diferentes sensações diretamente para a amígdala ou para áreas apropriadas do cérebro, como o córtex cerebral (KELLY, 1991; OHMAN; MINEKA, 2001).

3.3.2.4 Córtex sensorial

O córtex sensorial (Figura 23) é uma parte muito importante da área sensorial do cérebro, responsável pela análise e processamento dos sinais recebidos a partir do meio ambiente. O córtex sensorial distribui seus sinais de saída entre a amígdala e a região orbitofrontal (ARMONY; LEDOUX, 1997; ARBIB, 2002).

Figura 23 – Córtex motor e sensorial.



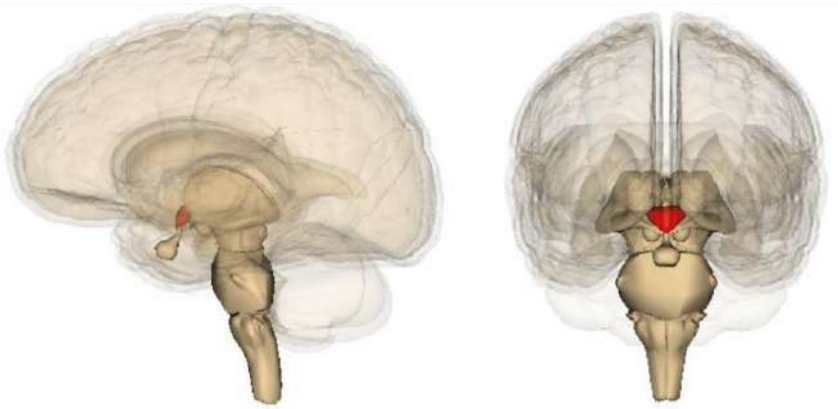
Fonte: adaptado de (TOLEDO, 2009).

De forma geral, a responsabilidade principal do córtex sensorial está relacionada com o recebimento dos sinais através do tálamo e, em seguida, realizar o processamento dessas informações para diversos fins. As informações recebidas a partir do córtex sensorial são extensivamente processadas. Desta forma, a amígdala e o córtex orbitofrontal recebem tais informações muito bem analisadas a partir do córtex sensorial (LEDOUX; FELLOUS, 1995; ROLLS, 1990)

3.3.2.5 Hipocampo e hipotálamo

O Hipotálamo (Figura 24) é um centro neural de bastante relevância para a manutenção da homeostase[6] do organismo. Além disso, é responsável por integrar várias respostas endócrinas, autonômicas e comportamentais que garantem a sobrevivência do indivíduo e a manutenção da espécie. Essas respostas estão envolvidas na regulação do metabolismo, fornecendo suprimento adequado de nutrientes e água do ambiente, permitindo a geração e cuidado com a prole, e defendendo o animal de predadores e outras ameaças (SCHACHTER, 1971).

Figura 24 – Hipotálamo.



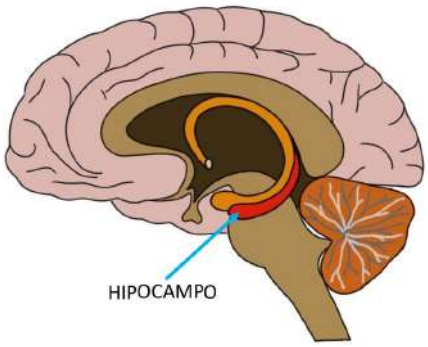
Fonte: adaptado de (ADAM, 2018).

A estrutura conhecida como hipocampo (Figura 25) é a porção alongada medial do córtex temporal (GUYTON; HALL, 2006). O hipocampo desempenha diferente funções, incluindo a navegação espacial, estabelecendo a memória de longo prazo e a formação das representações contextuais. Além disso, exerce uma forte ligação com a amígdala. De acordo a teoria do mapa cognitivo[7], o hipocampo é a estrutura cerebral relacionada com o mapeamento do ambiente (O'KEEFE; NADEL, 1978).

[6] Processo biológico de regulação que mantém o organismo em constante equilíbrio.

[7] A teoria sustenta que os animais podem aprender sobre as relações espaciais de objetos e relacionar eventos com o contexto espacial de sua ocorrência em seu sistema nervoso central.

Figura 25 – Hipocampo.



Fonte: adaptado de (ADAM, 2018).

3.3.3 Modelagem matemática do sistema límbico

Baseado no funcionamento e na forma estrutural do sistema límbico, foi possível desenvolver um modelo computacional na engenharia de controle que aproveitasse as características desse sistema, de forma a retratar o aprendizado emocional, ou aproximá-lo, em aplicações de controle com dinâmicas complexas. Esse modelo computacional é o resultado de um longo estudo e ensaios laboratoriais, todavia, o modelo computacional do sistema límbico não engloba todas as estruturas presentes do sistema, mas sistematiza as principais áreas que são essenciais ao funcionamento final: amígdala, córtex orbitofrontal, córtex sensorial e tálamo. Desta forma, considera-se esse modelo uma aproximação razoável do mecanismo do aprendizado emocional que ocorre na região do sistema límbico do cérebro. As duas primeiras estruturas (amígdala e córtex orbitofrontal) têm uma fundamental importância na questão do processamento das emoções, enquanto as outras, geralmente, funcionam como pré-processadores da entrada sensorial.

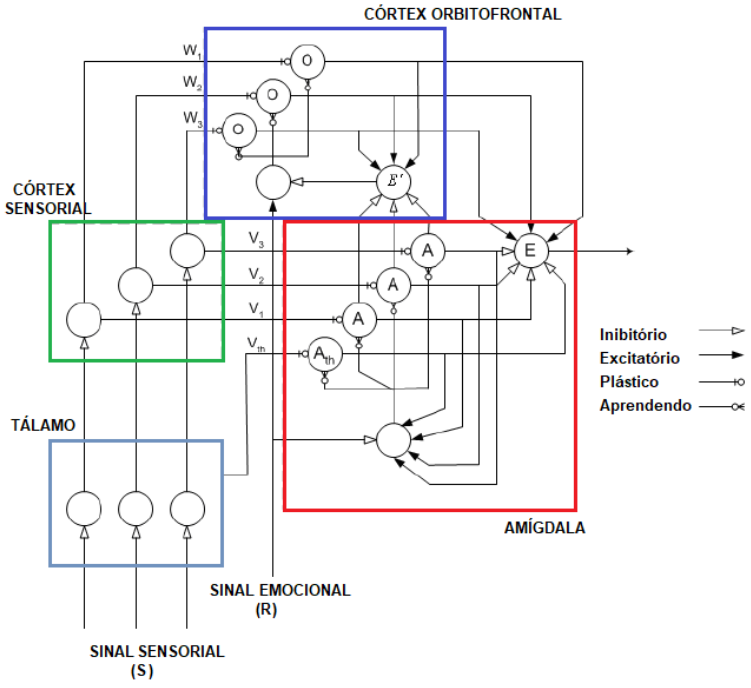
De acordo com esse modelo, a escolha da resposta final do sistema e a respectiva motivação para realizar essa resposta são diferentes, ou seja, a avaliação do estímulo e a escolha da ação são separadas (AGGLETON, 1992). Tal fato proporciona uma gama diversificada nos padrões de respostas provenientes dos estímulos. Tal característica advém da função biológica do processo de aprendizagem emocional do sistema límbico, onde a amígdala tem a função de aprender a associação entre a entrada sensorial e emocional, de maneira a evidenciá-la na saída do modelo (ROLLS, 1990; LEDOUX; FELLOUS, 1995).

A amígdala tem uma característica de aprendizado sempre crescente, ou seja, a amígdala é uma função monotônica. Nesse sentido, caso ocorra uma experiência, seja ela positiva ou negativa do ponto de vista da recompensa, a função da amígdala é capturar a essência dessa associação e utilizá-la como uma referência para as próximas experiências. Todavia, a saída final do modelo (ação de controle) ainda precisa ser mediada pela estrutura do córtex orbitofrontal.

Além disso, existe um caminho alternativo para o fluxo dos sinais de estímulos, que ao invés de passar pela córtex sensorial, acarretando uma demora devido ao tempo de processamento dessa estrutura, vai direto para a amígdala a partir do tálamo. Como resultado desse caminho, produz-se uma ação mais rápida, não ideal, conhecida como *decisão satisfatória*. Essa ação é uma amostra do sinal bruto, não processado no cortéx sensorial, servindo como uma prévia do sinal dos estímulos.

A Figura 26 apresenta o modelo computacional proposto por Morén (MORéN, 2002; MORéN; BALKENIUS, 2000) para o funcionamento do sistema límbico, que apesar de ser um modelo relativamente simples tem resultados experimentais satisfatórios, referindo-se à representação dos efeitos cerebrais das emoções. O sistema é formado por duas entradas principais. A primeira entrada remete às características sensoriais - *sinal sensorial* (S), a qual é proveniente de um estímulo externo, retratando o sistema sensorial (olfato, tato, visão, etc). No caso da segunda entrada do modelo, esta retrata o sinal emocional, ou a recompensa - *reward* (R).

Figura 26 – Modelo computacional do aprendizado emocional do sistema límbico.



Fonte: adaptado de (MORéN, 2002).

O funcionamento desse modelo faz uso de quatro estruturas principais, o tálamo, córtex sensorial, amígdala e córtex orbitofrontal. O tálamo recebe primeiramente os sinais da entrada sensoriais (S_i), realizando um processamento rápido e, logo após esse pré-processamento, envia o sinal para o córtex sensorial e também para a amígdala. No módulo do córtex sensorial, ocorre uma avaliação superficial e uma subdivisão do sinal proveniente do tálamo, enviando-o para a amígdala e córtex orbitofrontal (AGGLETON, 1992). No módulo da amígdala, o sinal deverá

sofrer uma avaliação "emocional" dos estímulos, sendo essa avaliação utilizada como referência dos estados emocionais. Na etapa final, a estrutura do córtex orbitofrontal tem a função de realizar a inibição das respostas consideradas inapropriadas provenientes da região da amígdala (MORéN, 2002).

De forma a obter o sistema de equações para a estrutura do modelo computacional apresentado na Figura 26, adotou-se para cada entrada do sinal sensorial do modelo um sinal S_i, onde *i* representa o i-ésimo sinal da entrada sensorial. A partir disso, para cada sinal de entrada sensorial existe um nó na amígdala (A_i) e no córtex orbitofrontal (O_i). Para cada uma dessas estruturas, os seus respectivos sinais de saída são obtidos através do produto entre o valor da entrada sensorial e o respectivo peso adaptativo, seja amígdala (V) ou córtex orbitofrontal (W), resultando em

$$A_i = S_i V, \tag{23}$$

$$O_i = S_i W. \tag{24}$$

No que diz respeito à estrutura do tálamo, sua característica é prover uma resposta rápida aos estímulos sensoriais, mesmo que tal resposta não seja ótima. Dessa maneira, nessa modelagem considera-se que o tálamo providencia um sinal máximo (A_{th}) entre todos os sinais recebidos pela entrada sensorial, enviado-o diretamente para a amígdala, podendo ser escrito como

$$A_{th} = max(S_i). \tag{25}$$

A etapa de aprendizado desse modelo, ocorre no processo da atualização dos pesos da amígdala e do córtex orbitofrontal, os quais atuam diretamente sobre o sinal sensorial *S*. As regras para cada atualização desses parâmetros são definidas como:

$$\Delta V = \alpha S_i max(0, R - \sum_i A_i), \tag{26}$$

$$\Delta W = \beta S_i R_o. \tag{27}$$

Os termos α e β são valores constantes e, geralmente distintos, entre zero e um, os quais fornecem um parâmetro para o controle da taxa de aprendizado da amígdala e do córtex orbritofrontal, respectivamente. No caso da atualização do peso da amígdala (ΔV), observa-se em (26) que este valor nunca pode ser negativo, ou seja, o valor absoluto do V é sempre crescente. Esta fato é condizente, pois uma vez aprendida uma reação emocional, esta deverá ser permanente, sendo apenas prerrogativa do córtex orbitofrontal realizar sua inibição, caso

esta seja inadequada. Por outro lado, o peso do córtex orbitofrontal W tem regra de atualização semelhante ao caso da amígdala, diferenciando-se no tocante que nesse caso, pode-se aumentar e diminuir conforme necessário, de forma a realizar a inibição necessária do sinal emocional.

O termo R_o em (27) é conhecido como *reforço interno para o córtex orbitofrontal*, podendo assumir dois valores distintos, dependendo do sinal emocional (*R*) existir ou não. A formulação de R_o pode ser escrita a partir da seguinte relação:

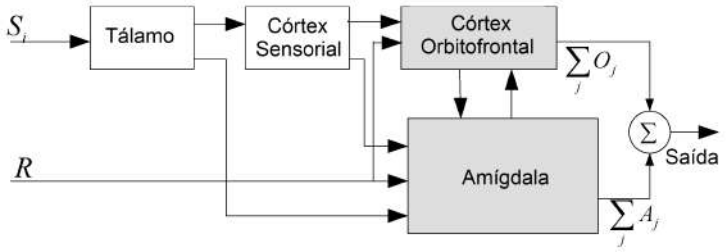
$$R_o = \begin{cases} max(0, \sum_i A_i - R) - \sum_i O_i & \forall \quad R \neq 0 \\ max(0, \sum_i A_i - \sum_i O_i) & \forall \quad R = 0 \end{cases} \tag{28}$$

A partir do momento de existência de uma recompensa no sistema, R_o representa a diferença entre a recompensa e as saídas da amígdala, subtraída da saída do córtex orbitofrontal. Por outro lado, caso não exista recompensa alguma, o córtex tem um comportamento diferenciado, R_o será o excedente das saídas da amígdala sobre as saídas do córtex.

Ao fim do processo, a saída do modelo - *model output* (MO) resume-se na diferença entre os somatórios das saídas, provenientes da amígdala e aqueles advindos do córtex orbitofrontal. O modelo completo em diagramas de blocos do sistema é apresentado na Figura 27. A saída final (ação de controle) do modelo é definida por (29), formulada da seguinte forma:

$$MO = \sum_i A_i - \sum_i O_i. \tag{29}$$

Figura 27 – Estrutura do modelo emocional do cérebro baseada na amígdala.



Fonte: adaptado de (BEHESHTI; HASHIM, 2010).

3.3.4 Estímulos no controlador emocional

O controlador BELBIC trata-se de um sistema de controle inspirado no controle emocional dos animais. As vantagens de tais sistemas são uma motivação pertinente para seu estudo, modelagem e consequente aplicações em diversos ramos da engenharia (HAITH; W., 2013).

A base de todo o aspecto do aprendizado deste tipo de controlador é centrada no módulo de aprendizagem emocional (BEL), o qual (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI,

2004) em seus trabalhos relacionaram a possibilidade deste módulo possuir aplicações adequadas para a engenharia de controle. A partir disso, desenvolveu-se o controlador BELBIC com base na criação de uma entrada sensorial (*S*) e um sistema gerador de recompensas (*R*) para o módulo BEL.

No ramo da engenharia de controle de sistemas, os sinais *S* e *R* fornecem ao módulo BEL a capacidade de torná-lo perceptível às mudanças do sistema dinâmico e, além disso, contribuem para atingir os objetivos de controle. No que diz respeito ao projeto do controlador BELBIC, associa-se à velocidade e ao ganho da resposta dinâmica o *S*, por outro lado, o *R* está intimamente relacionado à dinâmica de desempenho deste controlador.

Na literatura, a composição de *S* e *R* fica a critério do projetista, a partir do conhecimento da dinâmica da planta e de ensaios experimentais, faz-se possível obter tais sinais sensorial e emocional. Nesse sentido, pode-se optar por um conjunto de diferentes arquiteturas para ambos os sinais, como por exemplo, as diferentes variáveis da malha de controle a qual o BELBIC está inserido.

De acordo com (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004), os sinais *S* e *R* podem ser definidos da seguinte forma geral:

$$R = \sum_{i=1}^{K} x_i r_i, \tag{30}$$

$$S = \sum_{j=K+1}^{L} x_j s_j, \tag{31}$$

onde *R* representa a composição do sinal emocional, *S* a composição do sinal sensorial, os termos x_i e x_j são os pesos associados que necessitam de ajustes para um desempenho satisfatório (SHARBAFI; LUCAS; DANESHVAR, 2010), *r* e *s* são os parâmetros ou variáveis sensíveis ao modelo dinâmico.

Nos trabalho de (SHAHMIRZADI, 2005; LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004), utilizou-se o BELBIC no controle de um sistema dinâmico simples de submarino. Nesse caso, os sinais *S* e *R* foram ajustados da seguinte forma:

$$S = x_1 y + x_2 \dot{y}, \tag{32}$$

$$R = x_3 e + x_4 u, \tag{33}$$

onde *y* representa a saída da planta, *e* o erro e *u* o sinal de controle do BELBIC.

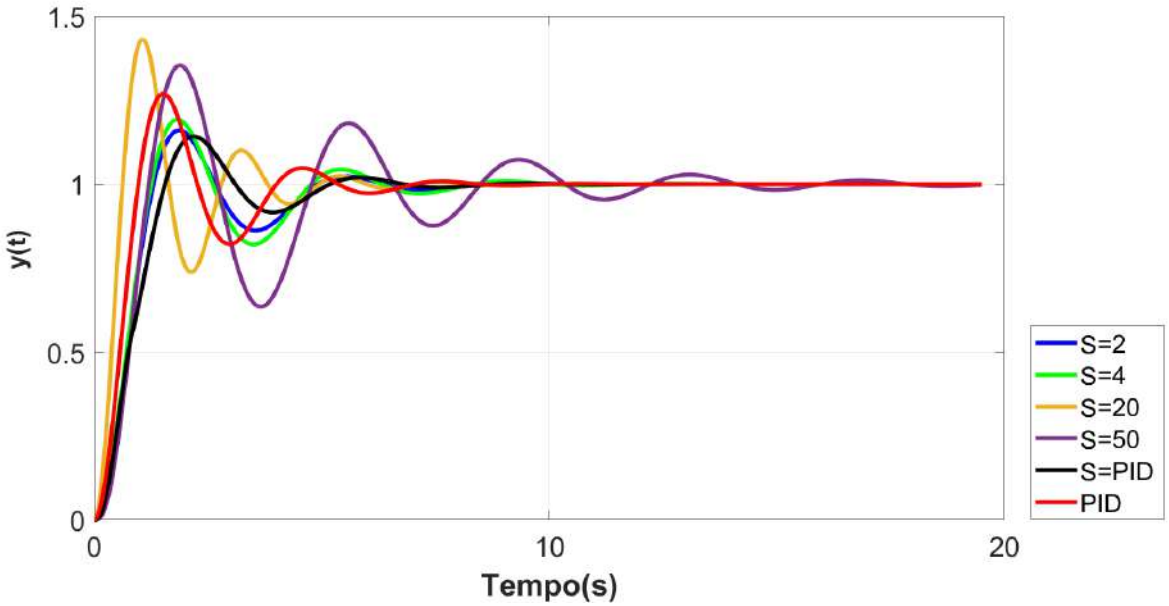
De acordo com trabalhos relacionados ao tema (DEHKORDI et al., 2011; MARKADEH et al., 2011; SHARBAFI; LUCAS; DANESHVAR, 2010; MARKADEH et al., 2011), nota-se que a definição adequada dos sinais sensorial e de recompensa, tal que possam promover o

correto e adequado funcionamento do BELBIC na engenharia de controle não é uma tarefa simples. Portanto, faz-se necessária a correta compreensão dos efeitos de ambos os sinais (*S* e *R*) no resultado final do controle do BELBIC e, além disso, as variáveis de controle da planta que formarão a arquitetura destes sinais. Todos esses aspectos tornam a modelagem desses sinais um processo iterativo, contendo diversos testes e ajustes para determinar os melhores valores atribuídos aos ganhos e, além disso, quais as variáveis de controle devem estar ou não envolvidas na composição dos sinais.

Neste sentido, o presente trabalho explora a possibilidade de se obter melhores desempenhos dinâmicos para o controlador BELBIC a partir da obtenção adequada dos sinais *S* e *R*. De forma a ilustrar esta possibilidade, utiliza-se como exemplo um controlador BELBIC em uma planta com uma dinâmica de segunda ordem. Neste caso, o controlador em questão é projetado utilizando como entrada para o sinal *R* um PID[8] e, referente ao sinal *S*, dispõe-se de valores distintos.

A Figura 28 apresenta o resultado da dinâmica de uma planta de segunda ordem controlada por BELBIC, o qual é projetado a partir de diferentes entradas de sinal sensorial.

Figura 28 – Dinâmica de resposta ao degrau unitário de uma planta de segunda ordem a partir do BELBIC com um PID em R e diferentes sinais para S.



Fonte: próprio autor.

Neste exemplo, compara-se a dinâmica da planta com um PID e também com um controlador BELBIC, o qual se utiliza do mesmo PID como um sinal para *R* e, diferentes sinais para *S* (constantes e o próprio PID). A partir da Figura 28 é possível notar que à medida que o

[8] $K_p = 0.1, K_i = 0.034$ e $K_d = 0.001$.

valor de S aumenta, a velocidade da dinâmica de resposta do controlador também aumenta. Por outro lado, quando S torna-se igual ao próprio PID, o efeito equivalente é de um sistema mais suave que o próprio PID, quando unicamente controlando a planta.

De fato, ao utilizar um controlador BELBIC com diferentes arquiteturas para os sinais S e R é possível obter desempenhos dinâmicos distintos. Neste caso, um projeto de controlador BELBIC deve necessariamente especificar adequadamente cada um destes sinais de acordo com a dinâmica da planta e os objetivos de controle.

3.4 Considerações finais

A estratégia de controle desenvolvida por (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004) permitiu criar sistemas de controle que fossem capazes de aproveitar, pelo menos em parte, as vantagens dos mecanismos dos processos com base no aprendizado emocional do sistema límbico. A lógica por trás da integração do modelo límbico em sistemas de controle de malha fechada pode ser comparada com a forma, aparentemente robusta, como o cérebro realiza a tomada de decisões. De fato, um sistema de controle tem relação direta com o processo de tomada de decisão: o objetivo do controlador é criar as melhores ações com base nas informações recebidas de acordo com os estados do sistema. Estas ações podem ser tomadas considerando o passado, o presente ou até mesmo previsões sobre os futuros estados do sistema.

4 CONTROLE EMOCIONAL BASEADO EM APRENDIZADO PROFUNDO

As abordagens de sistemas que buscam imitar o processamento emocional do cérebro têm chamado a atenção de diversos pesquisadores na área da AI nos últimos tempos (BABAIE; KARIMIZANDI; LUCAS, 2008).

4.1 Introdução

Algumas pequisas demonstraram, por exemplo, o desenvolvimento de novas ANNs, imitando alguns aspectos específicos da aprendizagem emocional, como no caso do hipocampo e amígdala (KUREMOTO et al., 2009a; KUREMOTO et al., 2009b). Além disso, vários modelos de predição baseados em emoções foram desenvolvidos para modelar sistemas complexos (PARSAPOOR; LUCAS; SETAYESHI, 2008). A grande maioria desses modelos possuem como referência o sistema amígdala-orbitofrontal proposto por Morén e Balkenius (BABAIE; KARIMIZANDI; LUCAS, 2008).

De forma geral, a pesquisa visa a adoção de técnicas recentes da área de DRL em conjunto ao modelo de controlador proposto em (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004), tal que seja possível obter uma nova arquitetura para este tipo de controlador, permitindo uma maior generalização para o controle de sistemas dinâmicos distintos.

4.2 Ferramentas

A partir dos objetivos deste trabalho, foi necessária a utilização de diversas ferramentas para a concepção, testes e utilização prática do modelo de controlador proposto.

No que diz respeito aos *softwares* e linguagens de programação utilizados neste trabalho, constatou-se a necessidade da utilização de diferentes tipos, principalmente devido à natureza dos objetivos propostos. Nesse sentido, de forma a realizar as simulações para a análise das respostas dinâmicas em tempo real dos sistemas dinâmicos, optou-se pela ferramenta computacional *Simulink*® (MATHWORKS, 2019). A motivação para esta escolha se deve, principalmente, às inúmeras interfaces visuais de fácil manipulação, o que acaba por facilitar a operação de simulação.

Na abordagem do uso das técnicas de DL, a linguagem de programação *Python*[1](FOUNDATION, 2020) é, sem dúvidas, a mais indicada para tal tarefa. Isso pode ser constatado nos diversos trabalhos que são publicados a respeito do tema (BEKLEMYSHEVA, 2015). A esmagadora maioria se utiliza de *Python*. O motivo disso deve-se à grande quantidade de *frameworks*[2] específicos de DL disponíveis. Além disso, grande quantidade de *datasets*[3] e

[1] Linguagem de programação de alto nível orientada a objetos.
[2] Ferramenta que une códigos comuns a diversos projetos.
[3] Conjunto de dados normalmente tabulados.

outras facilidades providas pela linguagem *Python* promovem a facilidade do trabalho nesta área da AI.

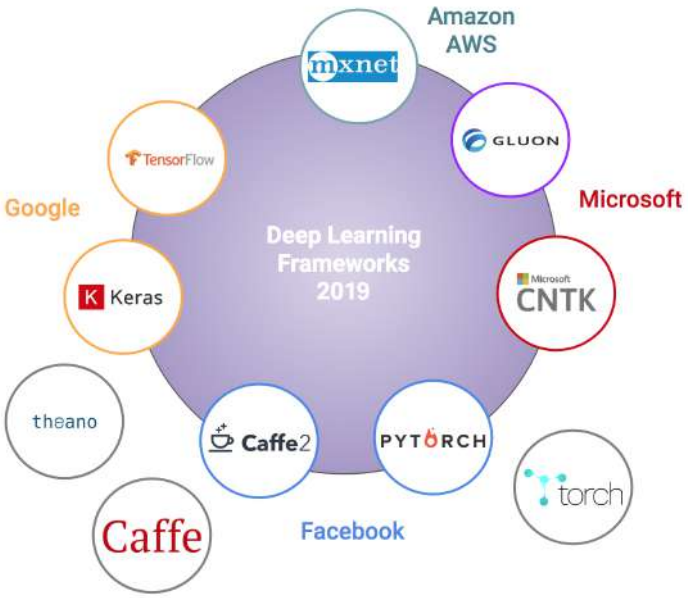
No caso da infraestrutura de *hardware* disponível, optou-se pela utilização de um computador dotado com uma placa dedicada para o processamento gráfico. O propósito desta ferramenta é proporcionar um melhor o treinamento das ANNs utilizadas neste trabalho. Além disso, com o propósito de implementação do controlador em um sistema embarcado para fins de aplicações práticas, optou-se por usar uma placa do tipo *Raspberry Pi* (BAUERMEISTER, 2019).

Por fim, um recurso importante para este trabalho é a infraestrutura de laboratório disponível. A partir desta estrutura, torna-se possível o estudo prático do controlador proposto neste trabalho. Desta forma, o laboratório possibilita um ambiente de análise e comparação adequada com outros métodos de controle, obtendo-se assim os resultados esperados neste estudo.

4.2.1 Framework TensorFlow

Atualmente, encontram-se disponíveis diversos *frameworks* que facilitam o trabalho com as técnicas de DL, como por exemplo o *TensorFlow, Theano, Keras, Pytorch*, entre outros (Figura 29). Neste trabalho, o *framework* utilizado para o projeto do controlador se concentra em um dos principais *frameworks* atuais, o *TensorFlow* (TF).

Figura 29 – *Frameworks* mais utilizados em DL.



Fonte: adaptado de (BAKKER, 2018).

O TF é uma biblioteca de *software* de código aberto para computação numérica usando

grafos[4] computacionais, desenvolvida por pesquisadores e engenheiros da equipe do *Google Brain Team*. O TF apresenta um nível de complexidade relativamente alto, porém é flexível e poderoso, sendo atualmente o principal *software* para desenvolvimento em DL em aplicações de AI (TENSORFLOW, 2019).

Uma das maiores razões pelas quais o TF é tão popular é o suporte a linguagens de programação para criar os modelos de DL. Atualmente, admite-se que *Python* é a linguagem mais conveniente para trabalhar com o TF, todavia, também existem interfaces experimentais disponíveis em outras linguagens como o *JavaScript, C#, C++, Java* e *Julia* (TENSORFLOW, 2019).

O TF opera com um gráfico de computação estática. De forma geral, o grafo é uma estrutura de dados que descreve completamente a computação que se deseja executar. Nesse sentido, primeiro define-se o gráfico, depois são executados os cálculos e, caso seja preciso realizar alterações na arquitetura, treina-se novamente o modelo. Além disso, as APIs[5] de alto nível do TF, em conjunto com os grafos computacionais, permitem um ambiente de desenvolvimento flexível e poderoso a partir dos recursos de produção disponíveis neste *framework*.

Uma característica importante do TF é a sua portabilidade, o grafo pode ser executado imediatamente ou mesmo salvo para uso posterior. Além disso, o grafo pode ser executado em várias plataformas: CPUs[6], GPUs[7], TPUs[8], bem como dispositivos móveis e embarcados. Por fim, ele pode ser implementado em produção sem depender de nenhum código que criou o grafo, apenas o tempo de execução necessário para executá-lo (TENSORFLOW, 2019).

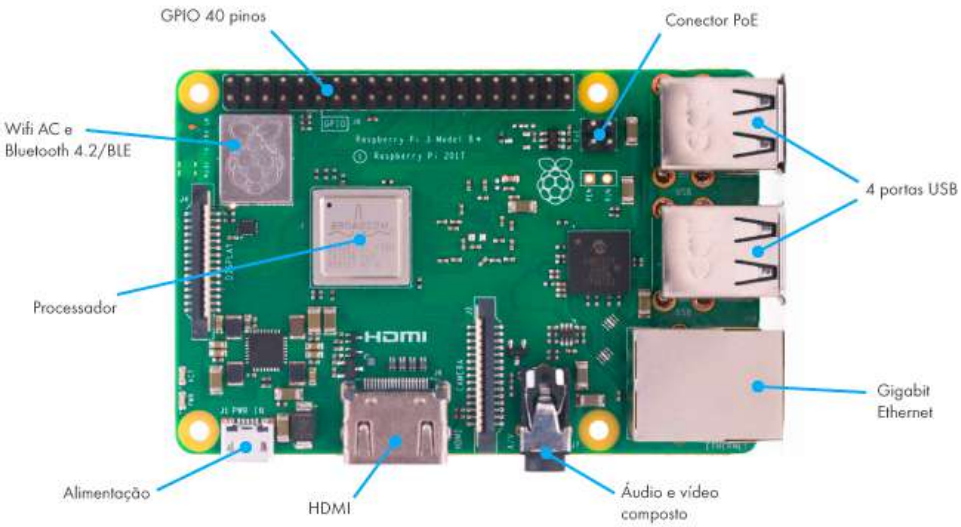
De forma geral, o TF é uma biblioteca de DL muito poderosa e madura, com fortes capacidades de visualização e várias opções para o desenvolvimento de modelos de alto nível. Por estas e outras razões de compatibilidade de uso prático, optou-se por utilizar o TF neste trabalho.

4.2.2 Raspberry Pi

De modo a tornar o controlador proposto uma aplicação prática real, compatível com as ferramentas de DL utilizadas e, além disso, permitir uma maior flexibilidade em seu funcionamento, analisaram-se várias opções de *hardware* para a prototipagem final da proposta deste trabalho. A partir de diversas opções de placas de processamento disponíveis, aquela que melhor atendeu às expectativas do presente trabalho foi a placa *Raspberry Pi*. A Figura 30 apresenta uma placa *Raspberry Pi* do modelo B+.

[4] Modelos matemáticos para resolver problemas práticos, representando relações entre objetos por meio de um conjunto de nós ligados por um conjunto de bordas ou arestas.
[5] Instruções e padrões de programação para acesso a um aplicativo ou software.
[6] Do inglês Central Process Unit ou Unidade Central de Processamento.
[7] Do inglês Graphics Processing Unit ou Unidade de Processamento Gráfico.
[8] Do inglês Tensor Processing Unit ou Unidade de Processamento de Tensor.

Figura 30 – *Raspberry pi* Modelo 3 B+.



Fonte: adaptado de (BAUERMEISTER, 2019).

De maneira geral, a placa *Raspberry Pi* pode ser considerada um minicomputador de baixo custo, compatível com sistemas operacionais, em sua maioria, baseados em *Linux*. Esta placa tem o tamanho aproximado de um cartão de crédito, possuindo diversos componentes que a tornam muito útil em diversas aplicações de sistemas embarcados.

As funcionalidades da placa *Raspberry Pi* são diversas, pois vem dotada de vários componentes e periféricos que proporcionam uma grande utilidade prática ao seu uso. Esta placa foi desenvolvida pela *The Raspberry Pi Foundation*, uma fundação educacional, a qual tem por objetivo promover a educação de adultos e crianças, em especial na área de computação e assuntos relacionados (RASPBERRY, 2019).

Em resumo, a adotação da placa *Raspberry Pi* neste trabalho deve-se à sua grande versatilidade em sistemas embarcados, tamanho reduzido com poder de processamento significativo, baixo custo associado, baixo consumo energético, entre outros aspectos. O modelo adotado para a utilização escolhido foi um dos mais recentes, o modelo 3 B+ (EICKHOLD, 2012; ESC, 2015).

A Tabela com as principais informações características referente ao modelo *Raspberry Pi* 3 B+ utilizadas no trabalho se encontra no Apêndice C.

4.2.3 GPU

Nos últimos anos, muitos dos avanços na área do DL foram possíveis graças ao desenvolvimento do processamento computacional. Atualmente, diversas soluções de DL baseiam-se, em grande parte, no uso das GPUs para acelerar e treinar aplicações complexas. Nesse sentido, as GPUs podem processar dados muito mais rapidamente do que as CPUs, pois trabalham de forma paralelizada, o que porventura reduz bastante o tempo de treinamento das ANNs. Por outro lado, o desenvolvimento de algoritmos para a utilização em GPUs requer um nível bem mais alto de sofisticação em programação, uma vez que apresentam um grau maior de complexidade (NVIDIA, 2019b).

Neste sentido, afim de viabilizar o processo de treinamento dos agentes de controle de DRL, optou-se por utilizar um computador com poder de processamento gráfico capaz de permitir a utilização das técnicas de DRL, utilizando-se do poder de processamento das GPUs de forma prática. Neste trabalho, o computador utilizado possui as configurações apresentadas na Tabela 2.

Tabela 2 – Características técnicas do computador utilizado neste trabalho.

CPU	Intel Core^{TM} i7-7700HQ quad-core Kaby Lake com clock de 2,8 GHz e 6 MB de cache
GPU	NVIDIA GeForce GTX 1050Ti com GDDR5 de 4 GB
RAM	16 GB DDR4
Armazenamento	SSD primário de 256 GB e HD de 1 TB
Rede	Wi-Fi802.11b/g/n/ac Bluetooth 4.0
Bateria	6 células de 74 Wh
Peso	2,62 kg
Dimensões	384 x 274 x 25 mm
Sistema Operacional	Windows 10

Por meio da NVIDIA GPU foi possível utilizar um recurso importante disponível pelo desenvolvedor, o CUDA *Toolkit*[9]. Esse conjunto de ferramentas permite o desenvolvimento, otimização e implementação de ANNs em sistemas embarcados acelerados por GPUs. De forma geral, CUDA pode ser definido como uma plataforma de computação paralela, desenvolvida com o intuito de permitir a exploração mais precisa e livre do grande potencial de processamento paralelo disponível por uma placa de vídeo (NVIDIA, 2019a). Além disso, utilizou-se nesse âmbito o *framework* cuDNN para acelerar a execução das ANNs na GPU da NVIDIA. O objetivo deste *framework* é fornecer uma biblioteca altamente ajustada para rotinas com padrão de funções comuns utilizadas em DL (NVIDIA, 2019c).

Todos esses recursos de processamento computacional demonstram-se muito importantes no âmbito dos treinamentos da DL, sendo utilizados de maneira geral em todo o presente trabalho.

[9] Conjunto de ferramentas, normalmente são implementados como uma biblioteca de rotinas ou uma plataforma para aplicativos que auxiliam uma tarefa.

4.2.4 Infraestrutura laboratorial

Uma parte importante deste trabalho visa a adoção prática do controlador proposto em um planta de controle real. O objetivo desta etapa visa a análise do comportamento dinâmico do controlador projetado e todas as possíveis situações pertinentes a um ambiente real. A fim de realizar este objetivo, o presente trabalho possui uma importante infraestrutura laboratorial à disposição, o *Laboratório de Eficiência Energética e Qualidade de Energia* - (LEEQE).

De forma geral, o LEEQE tem como missão realizar pesquisas com objetivos de eficiência energética em sistemas industriais. No presente trabalho, o laboratório é utilizado para a implementação prática do controlador proposto. A Figura 31 apresenta uma parte da infraestrutura disponível do laboratório.

Figura 31 – Painel de automação localizado no LEEQE.

Fonte: próprio autor.

Uma parte integrante do LEEQE é o *Laboratório de Otimização de Sistemas Motrizes* - (LAMOTRIZ). O LAMOTRIZ possui um ambiente necessário para a proposta prática deste trabalho, dispondo de diversos equipamentos de âmbito industrial e rede de comunicação interna própria. Além disso, neste ambiente, existem outros tipos de controladores atualmente em funcionamento, como é o caso do controlador tipo PID.

Por estas razões, este ambiente laboratorial tem a infraestrutura adequada para a análise e estudo da eficácia do controlador proposto. Além disso, permite o comparativo adequado com outros tipos de controladores reais.

4.3 Características de projeto do controle por reforço profundo

4.3.1 Agentes

Uma parte essencial deste trabalho é associar de forma adequada agentes de DRL ao controlador emocional. Dessa forma, propôs-se uma nova arquitetura para este tipo de controlador, permitindo uma melhora em seu desempenho dinâmico, sem a necessidade do exaustivo trabalho "braçal", referindo-se ao esforço na determinação adequada dos sinais sensorial e emocional.

Os agentes são compostos pelos algoritmos de DRL, os quais produzem os sinais de controle, ou seja, ações de comando baseadas nas informações advindas de um ambiente dinâmico. Estes algoritmos podem ser desenvolvidos de diversas formas e como consequência obter diferentes desempenhos, a depender da forma de como foram escritos. A fim de utilizar o estado da arte e, consequentemente, a melhor formulação dos algoritmos disponíveis atualmente, recorreu-se à biblioteca *OpenAI Baselines*.

A *OpenAI* é considerada uma instituição sem fins lucrativos, tendo por objetivo principal desenvolver pesquisas na área da AI. Sua missão é tornar, de certo modo, esse ramo da ciência "amigável", beneficiando a comunidade de pesquisadores ao redor do mundo. Este objetivo é realizado através da colaboração com outras instituições e com pesquisadores, tornando suas patentes e pesquisas acessíveis ao público como um todo (GERSHGORN, 2015; LEWONTIN, 2015).

Um dos resultados da *OpenAI* é a criação da biblioteca de código aberto *OpenAI Baselines*. Esta biblioteca reúne um conjunto de implementações de alta qualidade de algoritmos de DRL. Esses algoritmos facilitam as pesquisas, promovendo uma base sólida para diversas aplicações.

A Tabela 3 apresenta as características dos algoritmos de DRL presentes na biblioteca *OpenAI Baselines* (BROCKMAN et al., 2016).

Tabela 3 – Características dos algoritmos disponíveis na OpenAI Baselines.

Agente DRL	Tipo de observação		Tipo de ação		Suporte a
	Discreto	Contínuo	Discreto	Contínuo	redes recorrentes
A2C	√	√	√	√	√
ACER	√	√	√	⊘	√
ACKTR	√	√	√	√	√
DDPG	√	√	⊘	√	⊘
DQN	√	√	√	⊘	⊘
PPO	√	√	√	√	√
SAC	√	√	⊘	√	⊘
TD3	√	√	⊘	√	⊘
TRPO	√	√	√	√	⊘

Neste trabalho, são utilizados os principais agentes de DRL desta biblioteca. Cada agente possui especificações e hiperparâmetros únicos e, consequentemente, desempenhos diferentes. A proposta deste trabalho não está vinculada no comparativo entre os diferentes agentes de DRL, mas sim em sua utilização associada ao controlador emocional.

4.3.2 Ambientes dinâmicos

Na formulação dos problemas de DRL, a seleção do agente é fundamental. No entanto, por si só não é o suficiente, faz-se também necessário a construção adequada do ambiente ao qual este agente estará inserido. Uma vez selecionado o agente, determina-se a configuração e implementação do ambiente e, principalmente, a forma de comunicação entre o agente e este ambiente. O tipo de agente utilizado depende do modelo de ambiente construído.

Um ambiente em DRL é considerado como o meio pelo qual um agente realiza interações. Este ambiente recebe a ação por parte de um agente, a qual modifica ou não seu estado atual. Como consequência desta ação, o ambiente retorna um valor de recompensa ao agente, incentivando-o ou não a seguir uma determinada orientação de ações. Este ambiente pode representar situações distintas, como por exemplo, representar a dinâmica de um pêndulo invertido, contendo as equações que determinam as leis físicas para esta situação. Em outros casos, um ambiente pode ir desde um simples jogo de Atari, até complexos sistemas de controle dinâmicos.

Neste trabalho, alguns diferentes tipos de ambientes dinâmicos são utilizados a fim de promover a utilização dos agentes de DRL com o controlador emocional. Tais ambientes são formulados, em grande parte, por meio de simulações. Os ambientes utilizados neste trabalho são o *Simulink*®, a biblioteca *Python OpenAI Gym* e um protótipo de uma planta industrial.

4.3.2.1 Ambiente Simulink

De modo a facilitar a construção e consequente análise de sistemas dinâmicos de controle, resultantes da proposta do presente trabalho, optou-se pela utilização do *software Simulink*® para

a construção de ambientes dinâmicos. A motivação de tal escolha ocorre pelo fato do *Simulink*® ser uma poderosa ferramenta para modelagem, simulação e análise de sistemas dinâmicos lineares ou não-lineares, em tempo contínuo ou tempo discreto (MATHWORKS, 2019).

Uma vez determinadas as plataformas a serem utilizadas na construção dos agentes de DRL e dos respectivos ambientes dinâmicos, a próxima etapa concentra-se na determinação da interação entre ambos os sistemas. Neste caso, como o agente controlador é construído em uma plataforma distinta do ambiente em *Simulink*®, faz-se necessário determinar a melhor forma de comunicação entre eles. De fato, deve-se permitir a adequada e, necessária relação "ação-reação" entre o agente e o ambiente, referindo-se ao tempo de resposta do agente e atualização do estado dinâmico do ambiente.

Durante as etapas de pesquisa do presente trabalho, buscou-se testar vários tipos de comunicação a fim de obter aquele que melhor atendesse aos requisitos, ou seja, menor taxa de erro entre o fluxo de dados e boa taxa de atualização dos dados. Além disso, como o objetivo final é a utilização da proposta de controlador em ambiente real, a comunicação agente-ambiente em simulação deve levar em consideração as características de taxas de comunicação.

De modo inicial, apenas a critério de simulação computacional, uma das opções para para comunicação agente-ambiente foi a utilização de uma API específica do *MATLAB*® para o *Python* (*MATLAB*® *API for Python*). Esta API permite executar comandos do *MATLAB*® dentro do ambiente *Python* sem a necessidade da inicialização do *MATLAB*®. No entanto, apesar da facilidade de utilização desta forma de interação entre ambas as plataformas (*Python* e *Simulink*®), não foi possível utilizá-la nos treinamentos do agente de DRL. A razão disso deve-se ao fato de que o fluxo de informações, entre as plataformas *Python* e *Simulink*®, ocorre apenas ao final de cada simulação e, a despeito disso, o treinamento e simulação do agente necessita da troca de informações em tempo com o ambiente dinâmico.

Uma alternativa distinta de comunicação experimentada envolvia a utilização do protocolo de comunicação OPC[10], o qual é bastante utilizado em ambiente industrial. Todavia, durante a fase de testes e simulações, notou-se uma dificuldade na implementação prática, principalmente no tocante à disponibilidade de bibliotecas acessíveis e compatibilidade com as versões *Python* utilizadas. Por estas razões, tal protocolo de comunicação não foi adotado, no entanto, esse protocolo apresenta uma grande margem prática para a utilização em projetos futuros envolvendo esta área de estudo.

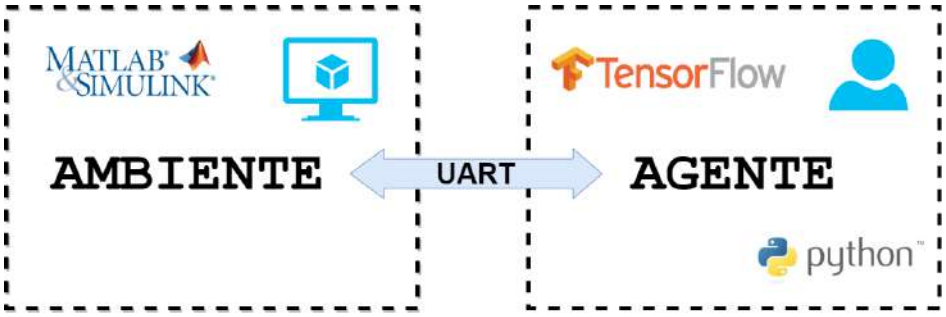
Por fim, a forma da comunicação entre as plataformas *Python* e *Simulink*® escolhida foi a serial com protocolo UART. Este protocolo de comunicação é utilizado por diversos micro-controladores, tendo como vantagem a simplicidade, além de menor custo quando comparado a outros tipos de protocolos (ROBOCORE, 2016).

[10] Abreviação do inglês OLE (Object Linking and Embedding) for Process Control ou Vinculação e Incorporação de Objetos para Processos de Controle.

A Figura 32 apresenta o esquema da conexão adotado entre as plataformas *Python* e *Simulink®*.

.

Figura 32 – Esquema agente-ambiente *Python* e *Simulink®*.

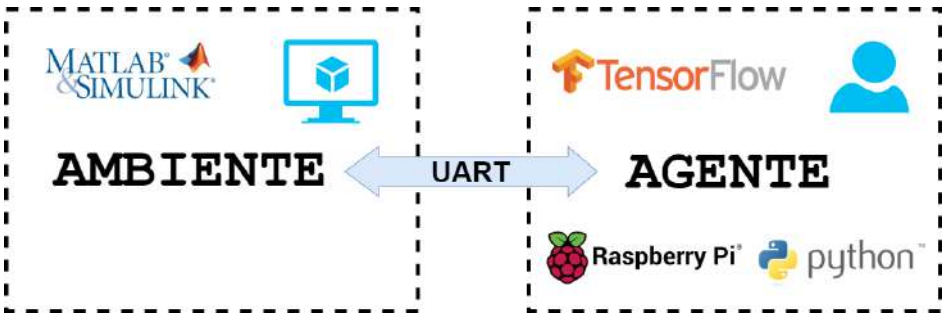


Fonte: próprio autor.

Uma vez que parte do objetivo do trabalho é o uso do controlador proposto remotamente, faz-se também necessário utilizar o *hardware Raspberry Pi* para o comissionamento da proposta. Da mesma forma anterior, o ambiente é construído via *Simulink®* e a comunicação com o agente controlador ocorre pelo protocolo serial UART, neste caso de forma física. Na prática, a utilização deste tipo de comunicação apresenta uma implementação amigável no tocante à realização de experimentos em ambiente laboratorial.

A Figura 33 apresenta o esquema da conexão adotado entre as plataformas *Python* e *Simulink®* via *Raspberry Pi*.

Figura 33 – Esquema agente-ambiente *Python* e *Simulink®* via *Raspberry Pi*.



Fonte: próprio autor.

4.3.2.2 Ambiente OpenAI Gym

Assim como os ambientes dinâmicos desenvolvidos a partir do *Simulink*®, a biblioteca *Python OpenAI Gym* oferece ambientes dinâmicos consistentes para a análise de controladores na área do DRL e semelhantes.

O *OpenAI Gym* (Figura 34) é um *toolkit* para o desenvolvimento e comparação de algoritmos de RL, o qual foi elaborado a partir dos trabalhos da (BROCKMAN et al., 2016). Esta ferramente oferece uma interface flexível de código aberto para ambientes de RL.

De forma geral, a comparação entre os diferentes algoritmos de RL é complexa, uma vez que as diferenças sutis na definição dos parâmetros do problema, como por exemplo, a definição da função de recompensa ou no conjunto de ações utilizados, alteram de forma significativa a natureza do problema, dificultando assim a reprodução e consequente comparação desses. Assim sendo, utilizar uma interface como a disponibilizada pelo *OpenAI Gym* padroniza a avaliação dos diferentes algoritmos de RL.

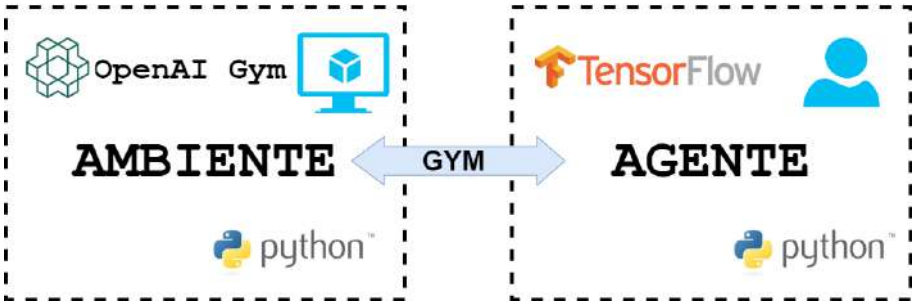
Figura 34 – Ambientes dinâmicos disponíveis do *OpenAI Gym*.

Fonte: adaptado de (BROCKMAN et al., 2016).

No caso específico do ambiente em *OpenAI Gym*, tanto o agente controlador como o ambiente dinâmico são desenvolvidos em *Python*, diminuindo assim a complexidade da análise do agente DRL, uma vez que não há a necessidade da configuração de comunicação como no caso do ambiente desenvolvido em *Simulink*®.

A Figura 35 apresenta o esquema da comunicação entre agente-ambiente no caso do ambiente desenvolvido no *toolkit OpenAI Gym*.

Figura 35 – Esquema agente-ambiente com o *OpenAI Gym*.



Fonte: próprio autor.

4.3.2.3 Ambiente planta industrial

Os ambientes dinâmicos abordados anteriormente se concentram em oferecer um meio pelo qual se faz possível realizar simulações, testes e análises referentes ao desenvolvimento da proposta de controlador neste trabalho. Por outro, o ambiente em um protótipo de uma planta industrial, no entanto, tem por objetivo associar o controlador proposto a uma situação prática de controle real. Por esta razão, vale salientar algumas das principais diferenças relevantes entre um ambiente simulado e um real.

Em primeiro lugar, referindo-se a um ambiente simulado, a velocidade das simulações pode ocorrer mais rapidamente do que o funcionamento do controle no caso real. Além disso, simulações podem ser realizadas de forma paralela, o que por sua vez acaba por acelerar todo o processo de aprendizado do algoritmo, o qual costuma ser lento.

Um ponto é que as simulações se utilizam de modelos de sistemas, os quais representam uma dinâmica real e, por mais bem elaborado que seja o modelo, este não é capaz de representar em sua totalidade um sistema real.

Por outro lado, referindo-se ao caso de uma situação real de controle, são vários os fatores que devem ser levados em consideração. Nesse sentido, destacam-se as limitações existentes de *hardware* e as condições de segurança do sistema, que devem ser adotadas para o funcionamento seguro do controlador. Diante disso, visando o correto e seguro funcionamento do processo, primeiro optou-se por utilizar um ambiente simulado para o treinamento do agente de controle e por fim, sua aplicação embarcada no protótipo da planta industrial.

A infraestrutura do laboratório neste trabalho dispõe de sistemas motrizes industriais, controlados e monitorados por meio de *Controladores Lógicos Programáveis* - (CLPs). Por esta razão, o controle da planta é realizado por meio de um sistema de controle digital. Com o objetivo de simular tal sistema, faz-se necessário modelar a planta através de um modelo de sistema discreto.

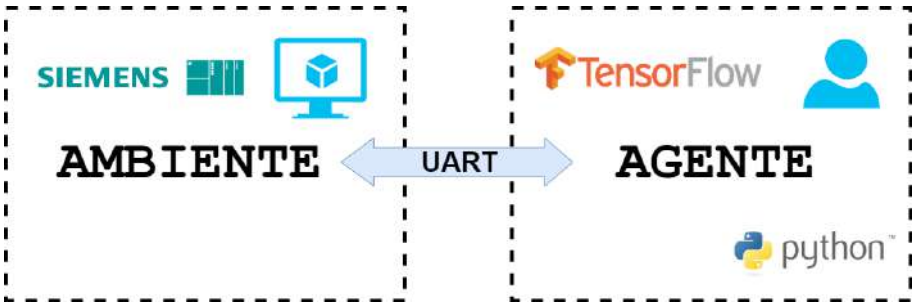
Geralmente, a forma mais simples da obtenção de um modelo de sistema dinâmico ocorre

na forma contínua (OGATA, 2003). A partir de tal situação, pode-se obter o modelo discreto do sistema, considerando-se o tempo de amostragem adequado para a representar as frequências de sua dinâmica. Neste trabalho, obtém-se o modelo discreto de um sistema motriz localizado no laboratório a partir do seu modelo contínuo, o qual foi previamente estabelecido por meio de experimentos prévios.

O modelo discreto da planta em laboratório é desenvolvido em ambiente *Simulink*®, a partir de seus recursos disponíveis. A partir de simulações, torna-se possível treinar, testar e adaptar o controlador proposto para atuar em um sistema de controle digital. Neste caso, o agente interage com o Simulnk, assim como em situações anteriores, contudo, neste caso específico o sistema está em modo discreto.

A Figura 36 demonstra o esquema que representa tal situação de interação agente-ambiente.

Figura 36 – Esquema agente-ambiente discreto com *Simulink*®.



Fonte: próprio autor.

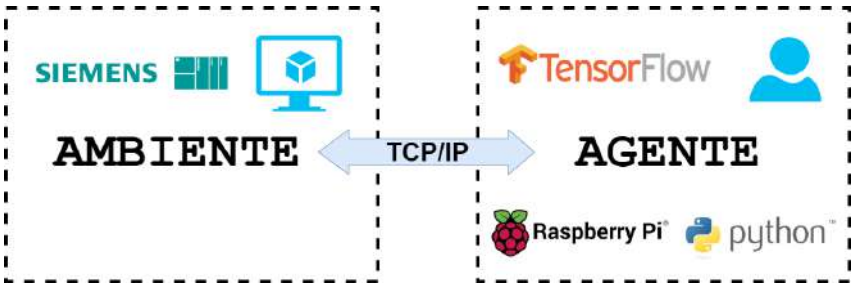
Logo que o controlador seja capaz de atuar em uma simulação de ambiente discreto, o mesmo pode então ser utilizado na bancada experimental. Por fim, após todos os testes necessários realizados, o controlador proposto é então embarcado na placa *Raspberry Pi*. Dessa forma é possível realizar o controle de um sistema dinâmico no laboratório (LEEQE) por meio do controlador embarcado, dedicado exclusivamente ao controle proposto. Neste caso em especial, a comunicação do controlador com o CLP é feito pelo protocolo TCP[11]/IP[12].

De forma a ter uma flexibilidade no uso do sistema de controle proposto, via *Raspberry Pi*, utiliza-se um sistema supervisório (WinCC) para o gerenciamento do sistema de controle desenvolvido.

A Figura 37 apresenta o esquema de comunicação agente-ambiente utilizado no caso do ambiente dinâmico da planta em laboratório.

[11] Do inglês Transmission Control Protocol ou Protocolo de Controle de Transmissão.
[12] Do inglês Internet Protocol ou Protocolo de Internet.

Figura 37 – Esquema agente-ambiente com *Raspberry Pi*-CLP.

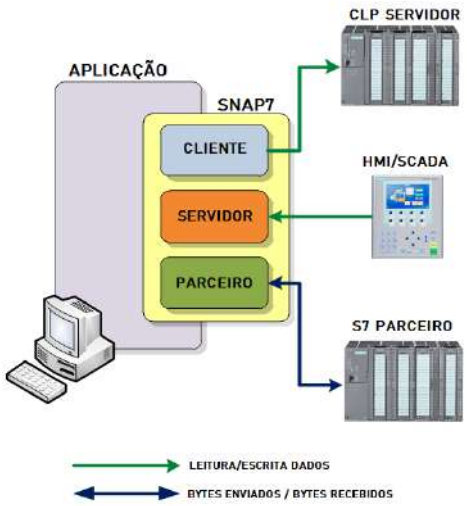


Fonte: próprio autor.

O recurso que permite a comunicação entre o ambiente da planta industrial e o agente de controle é o pacote *Snap7*[13] (NARDELLA, 2012). O Snap7 foi desenvolvido com o objetivo de superar algumas limitações dos servidores OPC, referindo-se ao processo de transferência de grandes quantidades de dados de alta velocidade em ambiente industrial. Nesse sentido, pode-se utilizar o Snap7 para aplicações embarcadas baseadas em Linux com arquitetura de processadores ARM[14] ou MIPS[15] como o *Raspberry Pi*, *BeagleBone Black*, *pcDuino*, *CubieBoard*, *UDOO* e *ARDUINO YUN*.

A Figura 38 ilustra o modelo de comunicação utilizado pelo pacote de comunicação Snap7.

Figura 38 – Modelo de comunicação Snap7.



Fonte: próprio autor.

[13] Pacote de comunicação Ethernet multiplataforma de código aberto de 32/64 bits para interface nativa com CLPs Siemens S7.
[14] Do inglês Advanced RISC Machine ou Máquina RISC avançada.
[15] Do inglês Microprocessor without interlocked pipeline stages ou Microprocessador sem estágios intertravados de pipeline.

De modo geral, os componentes principais se resumem em *cliente, servidor* e *parceiro*, os quais permitem realizar um interação entre seus sistemas baseados em uma cadeia de automação com CLP.

4.3.3 Recompensas necessárias

A etapa da definição das recompensas advindas do ambiente, resultantes das ações do agente atuante, é de relevante importância no comissionamento do controle com base em DRL. Essas recompensas, quando mal formuladas, resultam em situações não pretendidas pelo projetista. Além disso, a definição de tais recompensas é de certo modo irrestrita, pois varia de acordo com o problema abordado, o ambiente utilizado e a dinâmica que se deseja incentivar por parte do agente. Este aspecto, pode levar à criação das chamadas "recompensas escassas", ou seja, o objetivo que se deseja incentivar ocorre ao final de várias sequências de ações.

A escassez das recompensas pode levar um agente a realizar diversas ações aleatórias sem receber nenhuma recompensa em troca e, consequentemente, não obter nenhum aprendizado neste processo. Neste caso, dificilmente um agente encontrará a sequência adequada de ações que o levará à recompensa pretendida. Em outras palavras, o agente perde-se no meio do caminho.

A definição das recompensas neste trabalho apresenta tais preocupações. Por esse motivo, tem-se o cuidado de evitar torná-las muito escassas. Desta forma, busca-se incentivar o agente a aderir a um comportamento pretendido, sem a necessidade de realizar uma grande sequência de ações. Nos casos específicos dos sistemas dinâmicos que pretendem realizar um controle de rastreamento, o principal parâmetro utilizado na composição da função de recompensa é o erro, mais precisamente o distanciamento relativo do erro.

O erro pode ser visto como uma referência de qualidade na recompensa do agente de controle. O erro atesta ao controlador o quanto ele deve permanecer ou mudar o comportamento de atuação em um dado cenário. Portanto, utilizar-se do erro como uma medida avaliativa a cada instante de tempo contribui para um direcionamento mais preciso do agente. Neste caso, por exemplo, se em um dado instante de tempo o erro for muito excessivo, a recompensa recebida pelo agente neste momento pode ser a mínima possível, promovendo uma mudança no direcionamento de suas ações. Por outro lado, caso o erro seja mínimo, a recompensa nesta situação pode ser máxima, o que garante ao agente que suas ações estão sendo assertivas naquele instante de tempo.

Todavia, em muitas situações, apenas o valor instantâneo do erro não incentiva com clareza a direção ou mudança na dinâmica que um agente deve realizar, pois seu valor exprime apenas o quão afastado está o resultado das ações do agente em relação ao seu objetivo, não expressando por exemplo, com qual velocidade está se afastando ou aproximando. Desta maneira, outras variáveis da engenharia de controle podem ser utilizadas de forma a melhorar o incentivo nas ações do agente.

4.4 Caracterização dos estímulos do controlador emocional por meio do aprendizado por reforço profundo

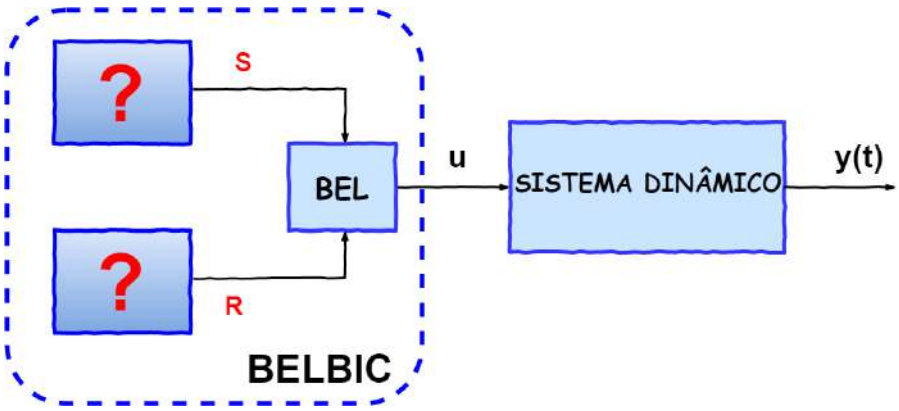
As ferramentas e características de projeto de controle em DRL apresentadas anteriormente nas seções 4.2 e 4.3, respectivamente, buscam fornecer as condições necessárias para o êxito da proposta de unir o controlador emocional às técnicas de DRL.

O controlador emocional pode representar uma alternativa viável para a utilização de sistemas de controle tradicionais, desde que seja corretamente arquitetado. Anteriormente neste trabalho, abordou-se a utilização do controlador por aprendizado emocional (seção 3.3), bem como suas implicações, principalmente no que tange a sua utilização na engenharia de controle (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004).

Embora o sistema de controle por aprendizado emocional tenha demonstrado muitos avanços em diversas aplicações (MARKADEH et al., 2011; SHARBAFI; LUCAS; DANESHVAR, 2010; SADEGHI; DARYABEIGI, 2014), uma questão relevante é a tratativa de seus sinais de estímulos (LOTFI; REZAEE, 2018). Em resumo, a questão central no funcionamento adequado deste tipo de controlador, concentra-se na definição adequada dos diferentes modelos para os sinais sensoriais e emocionais em aplicações distintas.

O projeto de controlador emocional não possui uma sistemática definida, devendo-se obter os parâmetros adequados para cada aplicação distinta. Nesse sentido, pode ser útil realizar uma associação deste controlador com um sistema ou algoritmo de autoajuste, o qual seja capaz de determinar de forma adaptativa os ganhos, pesos ou até mesmo a estrutura mais eficaz para os sinais emocionais e sensoriais do BELBIC. A Figura 39 ilustra a problemática da determinação dos estímulos do controlador emocional.

Figura 39 – Ilustração do problema de representação dos estímulos no BELBIC.



Fonte: próprio autor.

A proposta de associação do sistema de controle por aprendizado emocional com as técnicas de DRL é uma alternativa para auxiliar na melhoria do desenvolvimento e consequente

desempenho deste tipo de controlador. De forma semelhante a um sistema de controle por *feedback*, um agente de DRL pode ser projetado para atender aos requisitos de um sistema dinâmico de controle. Por meio de observações do estado de um ambiente dinâmico (variáveis da engenharia de controle), o agente realiza ações com o propósito de melhorar o desempenho dinâmico do sistema, corrigindo porventura alguns distúrbios e erros que possam vir a estar presentes nesta situação.

4.4.1 A função política do agente

De forma geral, a estrutura do agente de DRL responsável pelo mapeamento das observações em ações é a sua função política, em outras palavras, uma função com entrada e saída definida. A representação adequada da política de um agente permite traduzir de modo eficaz o relacionamento entrada-saída. Na área de DRL, a política pode ser vista de duas formas, *direta* ou *indireta*. Na forma direta, o mapeamento entre as observações e consequentes ações acontece de forma específica, uma observação implica diretamente uma ação. Por outro lado, a política indireta é aquela que se utiliza de algumas outras métricas, como por exemplo, a *Q-function*, a qual é utilizada para inferir uma saída.

A caracterização da função política pode ser uma tarefa complexa, pois em situações onde se exija uma grande quantidade de observações ou pares estados-ações, torna-se inviável representar esta situação por função matemática definida. Portanto, busca-se uma função tal que seja possível englobar uma maior representatividade, mesmo em situações com um alto grau de liberdade ou dinâmicas não lineares. Por estas e outras razões, opta-se pelo uso de ANNs profundas - *deep neural networks* (DNNs) para realizar a modelagem de tal política do agente.

Assim sendo, em vez de buscar-se por alguma estrutura de função não linear, capaz de funcionar em um ambiente específico, utiliza-se uma DNN, a qual se utiliza de uma combinação interna de nós e conexões para adaptar-se a ambientes distintos. Em outras palavras, como a política é uma função complexa, o objetivo das DNNs se resume em receber uma grande quantidade de observações e, a partir disso, traduzi-las em ações. Nesse caso, o agente de DRL aprende a função política à medida em que se interage com o sistema dinâmico.

A utilização de um agente baseado em uma política estruturada com DNNs, torna-se então uma alternativa importante para a caracterização dos estímulos pertinentes ao controlador emocional. Esta situação proporciona a capacidade de realizar-se uma modelagem não linear das funções que produzem tais sinais, produzindo-os com um maior grau de liberdade. A utilização de uma política nesse formato pode ser capaz de atender aos requisitos de funcionamento deste tipo de controlador, desde que seja corretamente formulada e treinada.

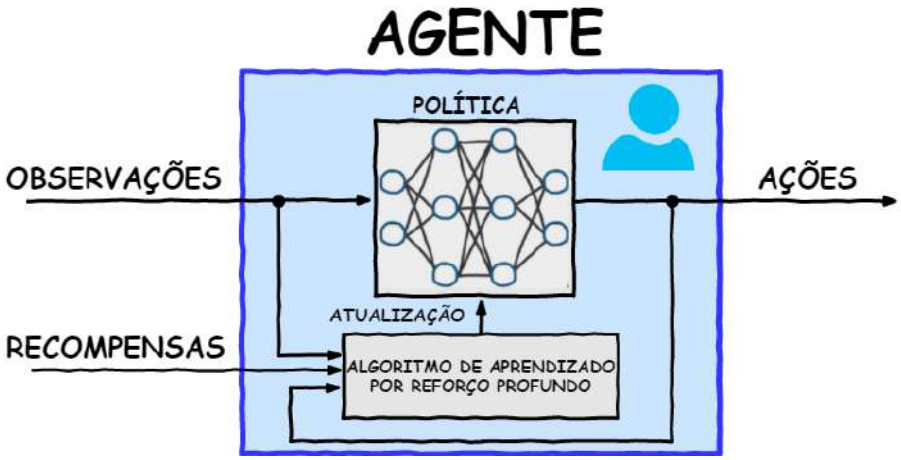
Embora uma política com base em DNNs possa ter uma natureza complexa, o comportamento universal de uma ANN garante que exista uma dada estrutura, tal que seja possível modelar perfeitamente o comportamento complexo de qualquer função não linear.

Um agente formulado com uma função política a partir de uma DNN pode ser considerado como um modelo caixa preta - *black-box*. Este modelo tem por característica a noção intuitiva de seu funcionamento e recursos ocultos, todavia, os valores reais ali dentro da "caixa" não são conhecidos de verdade. Nesse caso, deve-se considerar adequadamente este modelo, pois caso uma política não funcione como esperado, ou mesmo não atenda a alguma condição operacional específica do ambiente, dificilmente tem-se a noção exata do que modificar para ajustá-la, diferentemente do que acontece em sistemas de controle *feedbacks* tradicionais.

Além disso, deve-se atentar para o fato de que existem diversas estruturas distintas de DNNs, as quais podem não representar a complexidade necessária de algum tipo de problema. Portanto, faz-se necessário obter um conhecimento prévio da estrutura a ser utilizada, minimizando assim possíveis problemas na obtenção dos resultados. Tal estrutura de rede deve ser complexa o suficiente de maneira que seja possível aproximar a função desejada, porém, ao mesmo tempo não deve ser tão complexa ao ponto de tornar o seu treinamento inviável.

A Figura 40 apresenta o esquema interno de um agente com uma política formado por uma DNN.

Figura 40 – Estrutura de um agente formado por uma política com DNN.



Fonte: próprio autor.

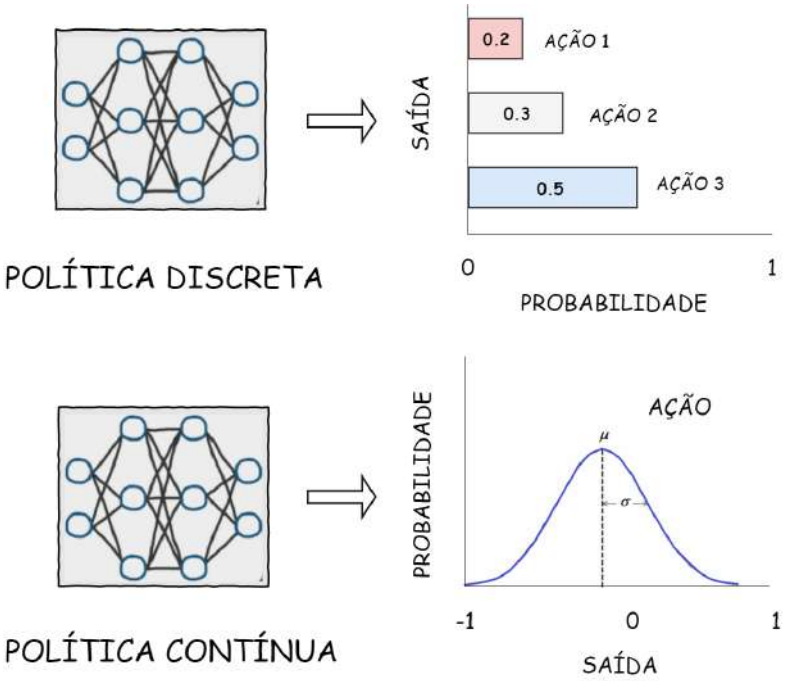
As políticas formadas por ANNs ou DNNs podem ser representadas distintamente, classificando-se através de seus espaços de ações, sejam eles *discretos* ou *contínuos*. As redes com políticas discretas geram funções do tipo *logits*[16] que por sua vez alimentam uma função tipo *softmáx*[17]. Tal condição é utilizada como uma distribuição de probabilidade para escolher uma determinada ação.

[16] Função que representa valores de probabilidade de 0 a 1 e infinito negativo para infinito.

[17] Função que recebe como entrada um vetor de K números reais e o normaliza em uma distribuição de probabilidade que consiste em K probabilidades proporcionais às exponenciais dos números de entrada.

Por outro lado, referindo-se às redes com políticas contínuas, em vez de probabilidades discretas, tomam-se como entradas pontos flutuantes entre -1 e 1 em uma distribuição normal. Neste caso, a política tem duas saídas ao invés de uma. A primeira saída é a média da distribuição de probabilidade (μ). Nessa situação, os valores amostrados devem ser centrados ao redor desta média. A segunda saída é o desvio padrão (σ), indicando o quão longe do centro está a média dos valores amostrados. A Figura 41 ilustra a diferença de resultados entre as políticas discretas e contínuas.

Figura 41 – Ações de um agente a partir de diferentes tipos de políticas.



Fonte: próprio autor.

A escolha do tipo do agente é de grande importância, pois a depender da arquitetura dos sinais sensoriais e emocionais do controlador emocional, um tipo de agente é mais indicado ou não.

Nesse contexto, um agente discreto pode ser indicado, por exemplo, na obtenção de ganhos fixos ou até mesmo na definição alguma arquitetura de sinal. Em contrapartida, um agente contínuo pode ser utilizado para a obtenção de ganhos variáveis que podem compor tais sinais.

Uma vez que neste trabalho optou-se por utilizar a biblioteca *OpenAI Baselines* (BROCKMAN et al., 2016) para a implementação dos agentes de DRL. As políticas limitam-se de acordo com a Tabela 4.

Tabela 4 – Políticas dos agentes de DRL disponíveis na biblioteca *OpenAI Baselines*.

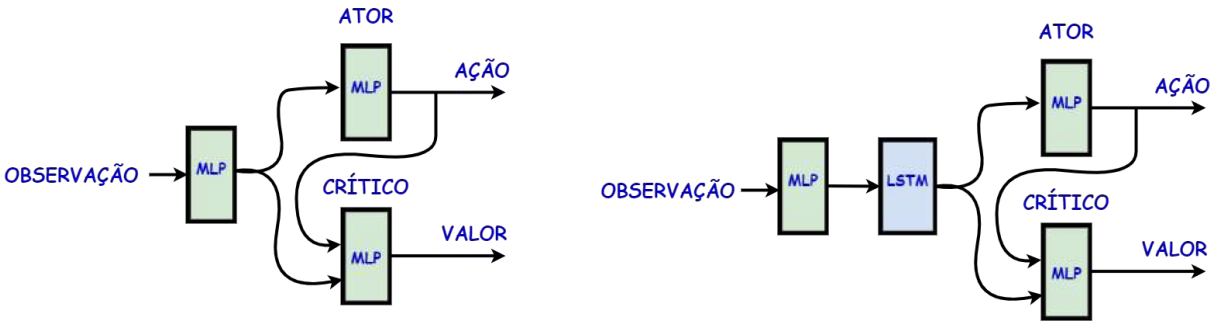
Arquitetura	Descrição do modelo
MlpPolicy	Política baseada em arquitetura AC usando redes MLP.
MlpLstmPolicy	Política baseada em arquitetura AC utilizando redes LSTM com uma rede MLP na extração de características.
MlpLnLstmPolicy	Política baseada em arquitetura AC utilizando redes LSTM normalizadas com uma rede MLP na extração de características.
CnnPolicy	Política baseada em arquitetura AC utilizando redes CNN.
CnnLstmPolicy	Política baseada em arquitetura AC utilizando redes LSTM com uma rede CNN na extração de características.
CnnLnLstmPolicy	Política baseada em arquitetura AC utilizando redes LSTM normalizadas com uma rede CNN na extração de características.

A Tabela 4 apresenta os diferentes tipos de implementações das políticas do *OpenAI Baselines* (BROCKMAN et al., 2016) e, cada tipo é apropriado para uma determinada característica de observação do ambiente. No que se refere aos problemas tratados neste trabalho, as políticas que envolvem redes convolucionais (*CnnPolicy*, *CnnLstmPolicy* e *CnnLnLstmPolicy*) na extração das características do ambiente não são aplicáveis, uma vez que tais políticas são essencialmente indicadas para o trabalho com processamento de imagens.

As políticas *MlpLnLstmPolicy* e *MlpLstmPolicy* implementam o AC utilizando redes LSTMs, as quais fazem uso do estado anterior para computar o próximo passo. Nesse sentido, a camada LSTM é compartilhada entre a rede MLP do ator e a rede MLP do crítico.

De modo geral, utiliza-se neste trabalho o modelo de políticas AC com redes do tipo MLP (*MlpPolicy*) e LSTM normalizada (*MlpLnLstmPolicy*) para a implementação dos agentes de DRL, os quais resultarão na composição dos estímulos no BELBIC. A escolha de tais tipos de políticas foi feita por meio de avaliações prévias de desempenho em treinamentos e simulações. A Figura 42 apresenta o modelo das políticas AC com redes do tipo MLP (*MlpPolicy*) e LSTM normalizada (*MlpLnLstmPolicy*).

Figura 42 – Arquitetura AC das políticas com modelos *MlpPolicy* e *MlpLnLstmPolicy*.



Fonte: próprio autor.

4.4.2 Arquitetura dos estímulos

A estrutura dos estímulos no controlador emocional pode assumir diversas formas de acordo com os objetivos de controle e a expertise do projetista. Geralmente, tais estruturas podem conter diferentes sinais associados a ganhos distintos, compondo dessa maneira uma arquitetura de estímulo capaz de produzir um sistema de controle emocional adequado. Dessa forma, pode ser muito desafiador o tratamento manual das diferentes arquiteturas individuais em cada estímulo.

Nesse âmbito, um ponto importante a ser mencionado é que, caso fosse possível utilizar um método na criação de uma política perfeita para um determinado ambiente, esta não seria "perfeita" o tempo todo. A razão deste fato é que normalmente um ambiente está em constante mudança dinâmica, ou seja, a criação de uma política estática não taria o desempenho ideal, o qual espera-se obter em algumas situações práticas nas quais existam a presença de distúrbios e ruídos inerentes ao ambiente.

De maneira a fornecer uma solução alternativa para auxiliar no desenvolvimento do controlador emocional, propõem-se diferentes arquiteturas de estímulos a partir do DRL. Nesse sentido, a estrutura proposta de controlador emocional denomina-se de *Deep Brain Emotional Learning Based Intelligent Controller* (DBELBIC).

Uma abordagem inicial do DBELBIC é a utilização de um agente de DRL capaz de sintetizar todo o sinal de estímulo em uma única função política em forma de *black box*. Diferentemente de tentar projetar cada componente de uma das funções dos estímulos, tudo se faz presente em uma única política. Tal função política recebe as observações do ambiente e produz diretamente os estímulos necessários para o controlador emocional.

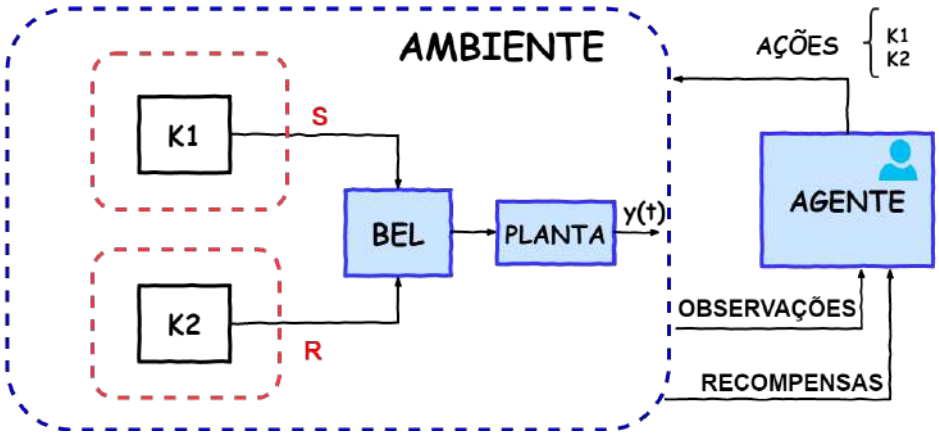
Em primeira análise, pode-se considerar de grande complexidade a construção de uma função *black box* que seja capaz de condensar toda a dinâmica dos sinais de estímulos, no entanto, o DRL é capaz de contribuir para essa situação.

Um agente de DRL, por outro lado, adapta sua política baseando-se nas ações realizadas, observações e recompensas recebidas do ambiente. A partir disso, o agente aprende a construir a melhor política à medida que interage com o ambiente. Assim sendo, um agente é capaz de adaptar-se ao ambiente independentemente da situação em que se encontra. Por esta razão, um estímulo criado a partir de um agente de DRL pode condicionar o controlador emocional para ser generalista o suficiente nas pequenas condições de mudança dinâmica de um ambiente, ou seja, capaz de rejeitar distúrbios e ruídos inerentes ao sistema.

Nesta abordagem inicial, o agente recebe as observações e recompensas advindas do ambiente dinâmico em questão, geralmente funções similares compostas por sinais comuns à engenharia de controle como o erro, a derivada do erro ou mesmo o sinal de *feedback* da planta. A partir disso, produzem-se de forma direta os sinais de estímulos sensorial e emocional. A Figura 43 ilustra a arquitetura desta proposta inicial de utilização do agente de DRL na determinação

direta dos estímulos do DBELBIC.

Figura 43 – Arquitetura DBELBIC tipo direta.



Fonte: próprio autor.

Nesta situação, as ações são as constantes *K1* e *K2*, representando o estímulo sensorial e emocional, respectivamente. Ambos os tipos de políticas podem ser utilizadas, discretas ou contínuas, dependendo de como se deseja produzir os estímulos. O agente discreto é utilizado para escolher algum ganho específico, limitado a certas constantes pré-definidas pelo projetista. Por outro lado, o agente com um espaço de ações contínuo, seleciona algum ganho a partir de um range específico de ações determinado previamente.

Embora a arquitetura direta dos estímulos por DRL possa ser conveniente, do ponto de vista do manuseio de sinais, este é um problema complexo para os algoritmos de DRL. Geralmente, tal situação é mais facilmente aplicada a sistemas de controle com ações restritas e objetivas. Nesse caso, cita-se como exemplo o controle do *swingingup*[18] de um pêndulo por meio de um agente discreto. Nessa situação as ações do agente podem ser restritas a três valores distintos, um torque máximo para direita, outro torque máximo para a esquerda e um torque nulo. Assim sendo, o agente tem uma maior percepção dos efeitos diretos de suas ações na dinâmica resultante do ambiente. No entanto, naqueles casos onde as ações do sistema de controle apresentam um maior grau de liberdade, ou até mesmo assumem valores muito distintos, dimensionar um sistema com ação direta pode ser um tanto difícil.

Nos casos em que envolvam problemas de seguimento da referência (precisão), a utilização de um agente para encontrar a ação direta (estímulos do controlador emocional) apresenta uma maior complexidade. Isso se deve ao fato dos vários critérios de controlabilidade e estabilidade, associados à engenharia de controle os quais o agente deve atender. Por esta e outras razões, esta abordagem inicial é restrita a alguns tipos de problemas de controle.

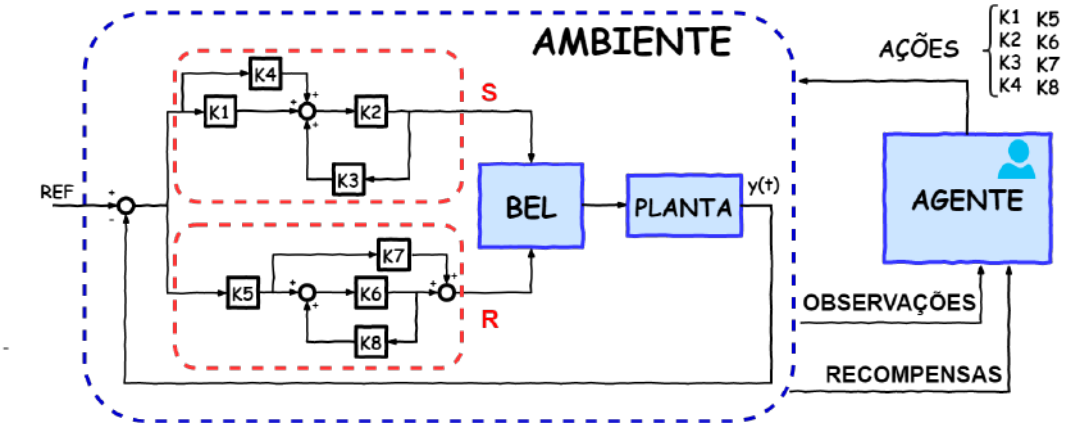
[18] Refere-se ao controle de balanço.

Uma possível alternativa na abordagem de utilização das técnicas de DRL em uma situação de controle mais complexa é a redução na escala do problema, restringindo-se o escopo de atuação do agente dentro de uma menor área de ação. Nesse caso, utiliza-se um agente de DRL para atuar no problema de forma especializada, diferentemente de tentar realizar ações em mais baixo nível a partir de observações em alto nível. Dessa maneira, reduz-se de forma considerável a função política do agente. Além disso, pode-se optar por utilizá-lo de forma específica em situações difíceis de se resolver com métodos tradicionais de otimização. Por fim, uma consequência direta dessa redução de complexidade é a diminuição no tempo de treinamento das redes.

Nesse contexto, a redução do escopo de atuação de um agente DRL também é aplicada a este trabalho. Nesse âmbito, existem diversas situações práticas, as quais um projetista tem a noção exata ou mesmo aproximada da arquitetura a ser utilizada em cada sinal de estímulo do controlador emocional, no entanto, os ganhos associadas às variáveis que compõem tais sinais precisam ser determinados a fim de obter uma melhor performance do controlador.

Nesse sentido, uma abordagem alternativa para a adoção de um agente de DRL junto ao controlador emocional é utilizá-lo na determinação de ganhos específicos, a partir de uma arquitetura de sinais previamente definida. Assim sendo, ao invés de tentar obter os estímulos individualmente, o agente pode obter os melhores ganhos simultaneamente que compõe tais sinais. A Figura 44 ilustra uma possível arquitetura baseada nesta abordagem denominada de indireta conhecida.

Figura 44 – Arquitetura DBELBIC tipo indireta conhecida.



Fonte: próprio autor.

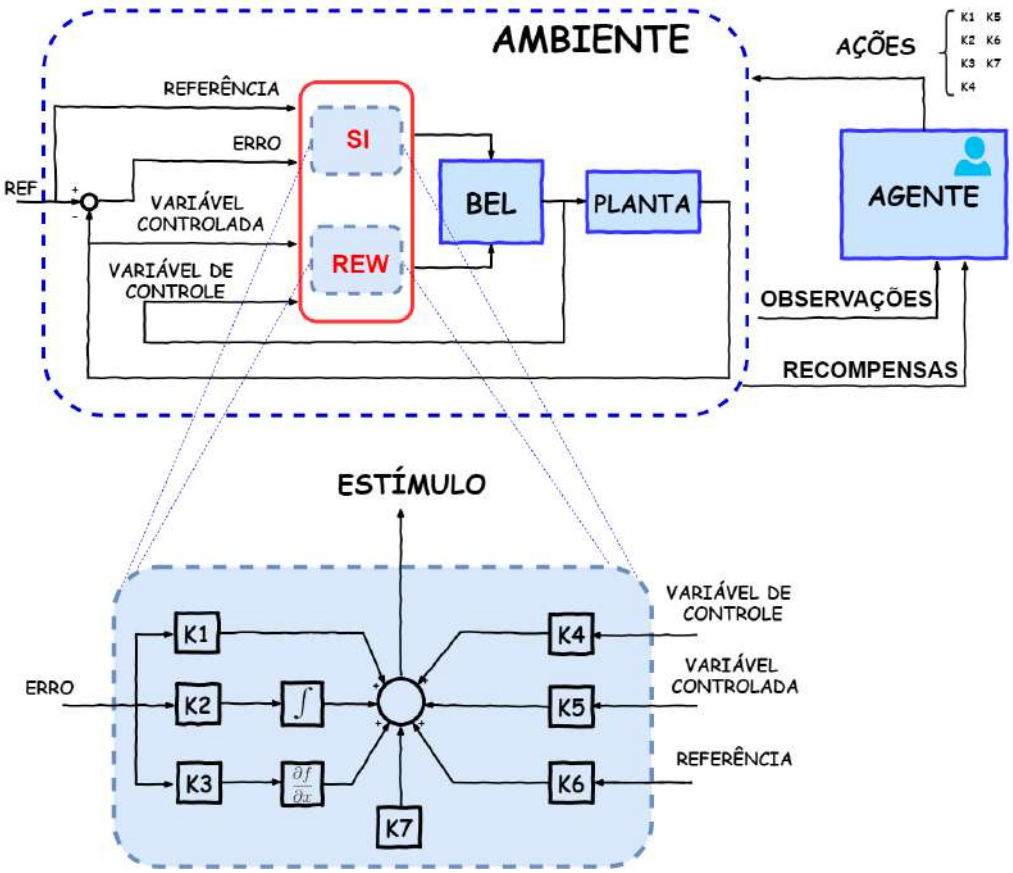
Neste caso, as ações do agente ficam restritas à obtenção de todos os ganhos K para cada constante do estímulo individualmente. Em relação a primeira abordagem (direta), esta situação representa uma diminuição na complexidade do agente, principalmente no que refere-se ao nível da ação, facilitando dessa maneira sua aplicação prática em sistemas reais. Nesta situação recomenda-se utilizar um agente do tipo contínuo, uma vez que a variação paramétrica entre

valores muito discrepantes pode afetar significativamente o modo de operação deste tipo de controlador emocional.

Todavia, apenas a diminuição da complexidade de atuação do agente de DRL por si só não resolve todo o problema, faz-se necessário verificar os critérios de controlabilidade e estabilidade necessários a um sistema de controle. Porém, diferentemente da abordagem inicial, subdividir o problema facilita possíveis intervenções manuais nos ganhos, ao invés de ter uma única caixa preta sintetizando todo o estímulo.

Por fim, uma abordagem final leva em consideração o não conhecimento prévio das arquiteturas dos sinais de estímulos do controlador emocional. Nesse caso, um agente de DRL é utilizado na definição da melhor arquitetura de tais sinais. A proposta leva em consideração os principais sinais disponíveis na planta. A Figura 45 apresenta a arquitetura da abordagem denominada de indireta desconhecida.

Figura 45 – Arquitetura DBELBIC tipo indireta desconhecida.



Fonte: próprio autor.

A partir da análise da Figura 45, nota-se que nesse caso um estímulo pode ser formado por diferentes sinais associados a ganhos K, os quais podem assumir dois valores distintos.

Esta abordagem pode ser associada conjuntamente a outras técnicas de otimização para a

determinação dos ganhos *K*. Dessa forma, inicialmente pode-se utilizar um agente de DRL para definir uma arquitetura básica e, a partir disso, otimizar tais ganhos dos estímulos por outras abordagens.

Em relação às propostas deste trabalho para a obtenção dos estímulos do controlador emocional, vale salientar que mesmo chegando a uma política eficaz em uma determinada condição de funcionamento, obtida por treinamento prévio, ainda assim é difícil prever com exatidão o comportamento do sistema em outra situação, baseando-se apenas em seu comportamento anterior. Esta análise é válida ainda que o sistema apresente um comportamento linear. Tal situação é explicada pelo fato de que em uma condição distinta, a ativação dos neurônios nas ANNs/DNNs ocorre de forma diferenciada, resultando possivelmente em uma condição não esperada de desempenho.

De forma a evitar ou minimizar esse risco, durante a etapa de treinamento, deve-se realizar outras combinações para a entrada do sistema de controle. Além disso, pode-se optar por realizar alterações, caso seja possível, nas condições iniciais problema. Dessa maneira, o sistema pode realizar mais observações e, consequentemente, adquirir uma maior capacidade de generalização das respostas em situações distintas. Este tipo de problema evidencia-se mais comumente na primeira abordagem, onde o agente é o responsável direto na elaboração do estímulo em um nível mais baixo.

Embora possa parecer sem necessidade, a realização de treinamentos extras deve ser levada em consideração. Uma vez que as DNNs possuem vantagem na tratativa de grande quantidade de dados, pode-se obter assim um melhor desempenho do sistema. Além disso, após os treinamentos feitos em ambiente de simulação, faz-se necessário realizar um "refinamento" do controlador proposto a partir de testes físicos, uma vez que o modelo em simulação não consegue ser perfeito em replicar todas as características reais do sistema.

Em resumo, a proposta do DBELBIC (utilização de técnicas de DRL para modelar diferentes arquiteturas dos estímulos no controlador emocional) se assemelha a outras técnicas da engenharia de controle, utilizando termos e procedimentos diferentes para representar os mesmos conceitos.

4.5 Análise de estabilidade

Na engenharia de controle, a estabilidade dos sistemas de controle por *feedback* é de fundamental importância, pois os sistemas dito instáveis não apresentam nenhuma utilidade prática. O objetivo principal da análise de estabilidade é o de realizar projetos de controladores adequados, capazes de atender especificações importantes de funcionamento, como rejeição a ruídos e distúrbios inerentes ao sistema (OGATA, 2003). O conceito formal de estabilidade auxilia na compreensão deste tema: *"um sistema é considerado estável se para toda entrada limitada ele produz uma saída limitada, não importa qual seja o seu estado inicial.* (OGATA, 2003; DORF;

BISHOP, 2018). De forma geral, tal definição implica dizer que para uma determinada entrada finita, o sistema não deve possuir saídas infinitas.

Nesse âmbito, a proposta do controlador deste trabalho necessita ser submetida a uma análise de estabilidade. Desta forma é possível avaliar se este controlador é capaz de respeitar as condições de estabilidade da engenharia de controle, assim como ocorre nos controladores tradicionais. No entanto, diferentemente do que acontece nas estruturas tradicionais de controladores, a abordagem para análise de estabilidade na área do controlador emocional apresenta algumas particularidades (dependendo da arquitetura dos sinais de estímulos tem-se uma malha de controle diferente). Além disso, como a proposta deste trabalho une o controlador emocional juntamente com as técnicas de DRL (DNNs), o sistema torna-se ainda mais não linear e portanto, aumentando assim sua complexidade.

Na área de sistemas não lineares existem diversos métodos capazes de abordar a questão da estabilidade. Um dos métodos mais famosos conhecidos é o de Lyapunov[19] (TSINIAS; KALOUPTSIDIS; BACCIOTTI, 1986). De forma geral, a teoria de Lyapunov pode ser dividida em dois métodos, o método *direto* e o *indireto*. No caso do método direto, busca-se associar funções escalares (funções de Lyapounov), as quais estão associadas com a energia do sistema e, a partir destas, verifica-se se a energia do sistema decresce ao longo do tempo. Por outro lado, no caso do método indireto, ocorre uma linearização em torno de um ponto de equilíbrio, permitindo investigar a estabilidade local de um sistema não linear através do seu modelo linearizado. Nesse último caso, os sistemas não lineares são aproximados através de uma série de Taylor[20] a partir dos pontos de equilíbrio, permitindo assim analisar sua estabilidade por meio de seus autovalores (GOLDHIRSCH; SULEM; ORSZAG, 1987).

No contexto do controlador emocional, o trabalho de (DASHTI; GHOLAMI; HAJIMANI, 2017) apresenta uma proposta para a estabilidade deste tipo de controlador com base em um grupo de sistemas lineares. Nos trabalhos de (KLECKER; PLAPPER, 2016; KLECKER; HICHRI; PLAPPER, 2017), são apresentados estudos para a estabilidade do controlador emocional, contudo, tais análises baseiam-se em um controlador de estrutura fixa, requerendo um conhecimento exato do sistema como um todo. Em (LOTFI; REZAEE, 2018), apresenta-se uma estrutura especifica de controlador emocional, denominada de *BELBIC Generalizado* (*G-BELBIC*) e, baseando-se nessa arquitetura, realiza-se uma análise de estabilidade por Lyapunov. Por outro lado, o recente trabalho de (KHORASHADIZADEH et al., 2019) apresenta um estudo da estabilidade do controlador emocional por Lyapunov, admitindo-se nesta situação uma estrutura de controlador emocional como um sistema não linear de aproximação universal.

A análise do controlador emocional a partir da perspectiva da teoria de Lyapunov é, sem dúvidas, um trabalho de grande relevância na área do estudo destes tipos de controladores inteligentes (KHORASHADIZADEH et al., 2019). Contudo, tais abordagens mencionadas

[19] Aleksandr Mikhailovich Lyapunov foi um matemático e físico russo (1857-1918).
[20] Brook Taylor foi um matemático britânico (1685 - 1731).

anteriormente neste trabalho levaram em consideração o fato de que as estruturas dos sinais emocionais e sensoriais eram devidamente conhecidas por parte do projetista e, a partir disso, tais sinais poderiam ser utilizados para compor a lei de controle do sistema em questão.

Diferentemente dessa situação, a abordagem do presente trabalho leva em consideração o fato de não haver o conhecimento da estrutura dos estímulos do controlador emocional, seja parcial ou total, o que torna necessário enfatizar essa perspectiva na análise em questão. Esta motivação se dá pelo fato das funções de Lyapunov serem utilizadas para realizar a análise de estabilidade quando ambos, o controlador e o sistema são conhecidos (PERKINS; BARTO, 2002; BOBITI; LAZAR, 2016).

4.5.1 Mapeamento da estabilidade

No que diz respeito a abordagem da estabilidade dos sistemas não lineares, em especial aqueles em que não é possível garantir uma convergência global, costuma-se considerar uma região específica de convergência. Essa região é delimitada de tal maneira que, dada uma trajetória inicial de um estado em seu interior, o percurso permanece totalmente em seu interior, convergindo por fim para um estado objetivo também dentro dessa área delimitada (KHALIL, 1996). Com base nessa perspectiva, (BERKENKAMP et al., 2017) propuseram uma análise de estabilidade de alguns sistemas dinâmicos cujas estimativas de estados são obtidas de forma sequencial a partir de processos gaussianos.

Apesar da teoria de Lyapunov apresentar ferramentas importantes para a análise da estabilidade de sistemas não lineares, diversos são os casos em que não é possível determinar uma convergência global do sistema (TSINIAS; KALOUPTSIDIS; BACCIOTTI, 1986). Nesse sentido, métodos de análise numéricos podem contribuir com o estudo da estabilidade destes tipos de sistemas. Uma vantagem importante no uso de uma análise numérica se concentra no fato de que praticamente toda não linearidade pode ser incorporada ao modelo de análise.

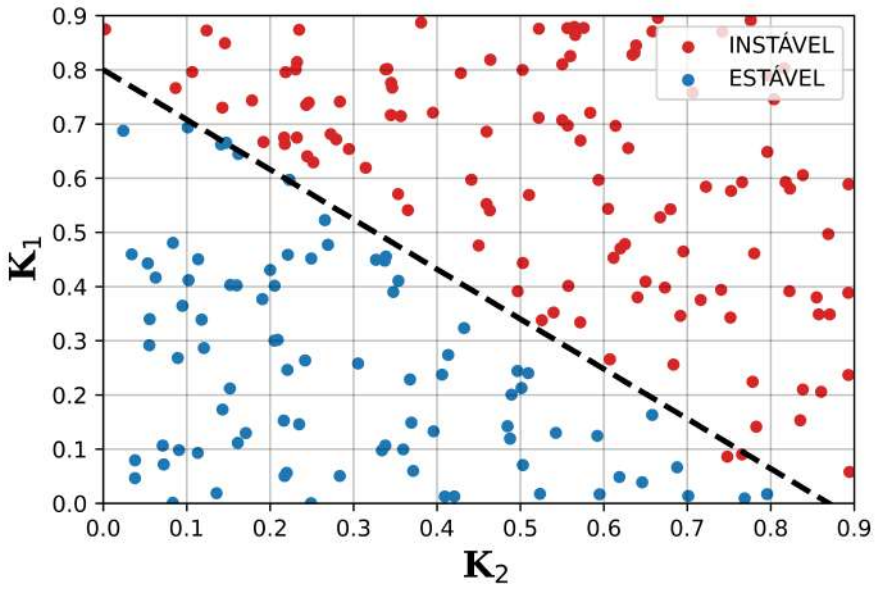
Neste trabalho, utiliza-se uma análise semelhante ao modelo numérico *Cell-to-Cell Mapping*[21] (HSU, 1987) para verificar a estabilidade da estrutura dos estímulos no controlador DBELBIC. Assim sendo, considerando a estrutura indireta de estímulos, propõe-se utilizar os agentes de DRL para realizar uma exploração inicial do espaço desses possíveis estímulos. Neste caso, o agente busca aleatoriamente testar diferentes combinações de estímulos, verificando assim qual faixa de estímulo torna o sistema instável.

Uma vez que o uso de técnicas que envolvem algoritmos de DRL podem demandar um tempo exaustivo de treinamento, a exploração inicial dos estímulos contribui para minimizar o espaço de busca das soluções, diminuindo assim a complexidade da tarefa de determinar os estímulos do controlador proposto. A Figura 46 ilustra um mapeamento linear de possíveis

[21] O método é baseado na discretização de uma parte do espaço de estados do sistema, definindo uma partição do espaço de estado em uma série de pequenas áreas (células). A partir disso, um mapeamento de célula a célula pode ser desenvolvido com base nas equações dinâmicas do sistema.

estímulos K_1 e K_2, obtidos a partir da amostragem de pontos aleatórios do controlador emocional formado por um ganho constante K_1 em *S* e outro ganho K_2 em *R*, aplicado em uma planta de segunda ordem.

Figura 46 – Exemplo de mapeamento de estímulos K_1 e K_2.



Fonte: próprio autor.

De forma geral, o mapeamento da estabilidade pode não fornecer toda a faixa de estabilidade do sistema, uma vez que depende de alguns fatores para seu êxito, como por exemplo, a quantidade de tempo de exploração prévia do agente de DRL e o espaço pré-definido de buscas. Todavia, este recurso apresenta uma contribuição evidente para a simplificação da construção de arquitetura dos estímulos no controlador DBELBIC.

A análise de estabilidade do controlador emocional por si só apresenta uma alta complexidade, principalmente por se tratar de um sistema de controle não linear (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004). No caso do presente trabalho, existe uma dificuldade ainda maior, pois a associação desse controlador com DNNs torna maior o grau de dificuldade de tal análise.

No que diz respeito a estabilidade de sistemas de controle, os quais se utilizam de RL, alguns trabalhos apresentaram comprovações matemáticas para o sucesso dessas aplicações (LILLICRAP et al., 2016; LI, 2017), em especial o trabalho de (JIN; LAVAEI, 2018), no qual aborda-se o problema da garantia de estabilidade das políticas de aprendizagem por reforço quando estas são conectadas com sistemas dinâmicos não lineares.

4.5.2 Estabilidade na perspectiva do aprendizado por reforço

O conhecimento a respeito do modelo do sistema é de grande importância para o projeto de controladores, principalmente os caracterizados como *tradicionais*, uma vez que se utilizam das informações da dinâmica do sistema para projetar seus ganhos específicos (OGATA, 2003). Além disso, esses projetos podem levar em consideração algumas incertezas do modelo, permitindo assim construir uma malha de controle capaz de estabilizar melhor o sistema, deixando-o mais robusto e menos susceptível aos ruídos externos. De modo distinto, grande parte das técnicas da AI não levam em consideração o modelo do sistema, como é o caso de alguns dos algoritmos de RL, os quais aprendem a dinâmica do sistema e atualizam seus ganhos a medida que interagem com este sistema (LANDAU et al., 2011; SUTTON, 1984).

No geral, a abordagem dos projetos de sistemas de controle baseados em RL apresentam uma maior exigência quando comparados com a abordagem para estruturas de controladores tradicionais, principalmente no que diz respeito às garantias de estabilidade, viabilidade e robustez (LANDAU et al., 2011). Além disso, como no caso dos agentes de DRL que se utilizam de DNNs para construir leis de controle não lineares, entender o comportamento de estabilidade e convergência para tal situação não é um assunto trivial (LEWIS; VRABIE; VAMVOUDAKIS, 2012; GÖRGES D., 2017).

No contexto dos sistemas de *feedback* a partir da utilização de controladores baseados em DRL, faz-se de grande importância determinar os meios pelos quais seja possível garantir a estabilidade do sistema, uma vez que os algoritmos tradicionais de DRL têm o objetivo principal voltado à performance do controlador (LILLICRAP et al., 2016; MNIH et al., 2013), em detrimento da "segurança" de suas ações. Além disso, as técnicas tradicionais de análise de estabilidade, em grande parte, não são eficazes nesta situação, principalmente devido à própria natureza do problema (GEIBEL; WYSOTZKI, 2005).

A respeito desse tema, o trabalho de (JIN; LAVAEI, 2018) apresentou uma importante análise a cerca da certificação da estabilidade dos sistemas de *feedback* baseados em controladores de RL. Nesse trabalho, demonstrou-se que, por meio da regulação dos gradientes da política do agente, é possível se obter fortes garantias para a estabilidade robusta do sistema. A análise sugere uma lei de controle do agente de DRL descrita da seguinte maneira

$$u(t) = \pi_t(y(t); \theta_t) + e(t), \tag{34}$$

onde π_t é ação resultante de uma política parametrizada por ANNs, a qual é variante no tempo devido ao aprendizado online do algoritmo. Adicionalmente a essa política, tem-se um termo de exploração $e(t) \in \mathbb{R}^{n_s}$, o qual tem por objetivo realizar a captura do efeito randômico do controle durante a fase de aprendizado (JIN; LAVAEI, 2018).

De forma geral, tal abordagem sugere a utilização das informações do gradiente de $\pi_t(y(t);\theta_t)$ para a certificação da estabilidade, uma vez que tais informações podem ser obtidas em tempo real de maneira simples e, além disso, podem ser consideradas genéricas o bastante de modo que possam englobar um grande número de controladores não lineares (JIN; LAVAEI, 2018).

O resultado principal desse trabalho relata que, caso existam valores limites de gradientes $(\xi, \bar{\xi})$, os quais satisfazem algumas condições demonstradas por (JIN; LAVAEI, 2018) em seus trabalhos, então o sistema interconectado apresenta um ganho L_2[22] finito, desde que $\pi_t \in \mathcal{P}$, onde $\mathcal{P}$ é descrito da seguinte forma

$$\mathcal{P}(\xi) = \left\{\pi \mid \xi_{i,j} \leq \partial_j \pi_i(y) \leq \bar{\xi}_{i,j}, \forall i \in [n_a], j \in [n_s], y \in \mathbb{R}^{n_s}\right\}, \tag{35}$$

onde n_s e n_a são os conjuntos de espaço do sistema e do controlador, respectivamente. A equação descrita em (35) apresenta um conjunto de controladores não lineares, os quais possuem derivadas parciais limitadas por ξ e $\bar{\xi}$. Nesse caso, deseja-se que a política do agente permaneça dentro desses limites de segurança de modo a garantir sua estabilidade.

De qualquer forma, garantir a estabilidade de controladores que envolvam sistemas de DRL é um desafio relevante. Além disso, as considerações a respeito de aproximação de funções não lineares e sua estabilidade são ainda mais complexas quando considera-se a dinâmica e as recompensas do sistema em tempo contínuo (LEWIS; VRABIE; VAMVOUDAKIS, 2012; JIANG Y., 2002)

4.5.3 Identificação do modelo

As análises de estabilidade para os sistemas não lineares, mencionadas neste trabalho, são de grande importância teórica para a aplicação no controlador proposto deste trabalho. Por outro lado, apesar de apresentarem uma abordagem matemática satisfatória, não se demonstram como ferramentas amigáveis de uso prático para a análise da estabilidade.

Nesse sentido, propôs-se como alternativa a essas análises formais a utilização de um modelo aproximado do sistema em forma de função de transferência, a partir de uma condição específica de funcionamento, obtendo dessa forma a possibilidade da utilização das ferramentas práticas de análise da estabilidade de sistemas *feedback*.

A partir da definição da estrutura do DBELBIC, busca-se realizar a modelagem desse sistema como uma única de função de transferência $G(s)$. Essa identificação consiste em perturbar o sistema, aplicando um sinal conhecido à sua entrada e, a partir disso, a resposta do sistema é observada no decorrer do tempo.

[22] Métrica para avaliação da estabilidade absoluta e estabilidade para sistemas com entrada e saída limitada .

De posse desse conjunto de informações (entrada e saída do sistema), ferramentas matemáticas são aplicadas, as quais são capazes de gerar um modelo dinâmico para o sistema analisado.

De maneira a obter um modelo condizente ao sistema original em análise, deve-se escolher adequadamente o tipo de pertubação a ser aplicada à entrada desse sistema. A escolha da característica de tal pertubação é importante, pois a depender do formato do sinal de excitação, o mapeamento da dinâmica do sistema em determinada região de frequência pode não ser eficaz (LJUNG, 1999).

Existem diferentes tipos de sinais que podem ser aplicados nessa tarefa, especialmente o sinal degrau, a onda quadrada e o sinal pseudo aleatório - *Pseudorandom binary sequence* (PRBS). O sinal mais clássico e simples é o degrau, o qual fornece uma resposta de fácil compreensão. No caso do sinal PRBS assumem-se dois valores distintos, os quais mudam a cada intervalo de tempo de forma determinística pseudoaleatória. Dessa maneira, o sinal PRBS é capaz de excitar adequadamente os modos de oscilação do sistema em uma faixa maior de frequência (LJUNG, 1999).

Nesse trabalho se fez uso de sinais do tipo degrau para a identificação do sistema. Uma vez obtida a modelagem $G(s)$ que melhor representa o sistema do DBELBIC, torna-se possível utilizar as técnicas práticas de análises da estabilidade, como no caso dos trabalhos de Bode[23] (OGATA, 2003).

4.6 O projeto do controlador

O uso das técnicas de DRL apresentam muita flexibilidade em diversas aplicações, ou seja, grande capacidade de adaptação às dinâmicas do ambiente, principalmente se processo de treinamento do agente de controle for bem executado (ARULKUMARAN et al., 2017; LI, 2017).

De maneira a elaborar um projeto adequado do controlador proposto neste trabalho, faz-se necessário compreender as variáveis e condições que envolvem a concepção do processo, permitindo executar o treinamento dos agentes de DRL adequadamente. Nesse sentido, busca-se observar os casos de sucesso, tanto no uso do controlador emocional quanto na utilização de ANNs/DNNs em sistemas de controle, tomando-os como referências paramétricas.

4.6.1 Características de comissionamento

No início de uma nova aplicação, dificilmente se tem o conhecimento exato de quais são as escolhas mais adequadas para o projeto, principalmente no que diz respeito ao tema que envolve as ANNs. De forma geral, o processo na prática é em grande parte um método empírico, ou seja, várias decisões precisam ser tomadas baseadas em análises dos experimentos realizados.

[23] Hendrik Wade Bode foi um engenheiro estadunidense (1905-1982).

No que se refere ao RL que se utiliza de ANNs/DNNs para construir a política dos agentes, o modelo inicial obtido após o treinamento da rede geralmente pode apresentar um comportamento indesejado. Isso pode acontecer devido ao fato de que o ambiente não foi explorado devidamente ou mesmo que o próprio treinamento não foi concebido de forma eficaz. Em tais circunstâncias, a depender dos resultados obtidos, algumas das possíveis soluções convenientes podem ser aumentar a profundidade da rede, treinar a rede por um tempo demasiadamente maior, testar diferentes algoritmos de aprendizado, experimentar novas arquiteturas de rede, acrescentar métodos de regularização, entre outros.

Nesse contexto, os algoritmos de DRL disponibilizados pela *OpenAI* apresentam uma implementação otimizada em relação ao *framework TensorFlow*, o que garante uma boa confiança no projeto do agente de controle em geral.

Uma etapa importante no dimensionamento de agentes de controle que se utilizam de DNNs é a definição adequada dos hiperparâmetros. De modo geral, os hiperparâmetros podem ser definidos como as variáveis que controlam o processo de treinamento e consequente aprendizagem da rede.

O dimensionamento de uma ANN/DNN passa por definir a quantidade de camadas ocultas, quantidades de unidades por camada, taxa de aprendizagem, funções de ativação, tamanho do *mini-batch*[24] e assim por diante. Essas variáveis são de configuração do treinamento e não estão relacionadas ao dados do treinamento.

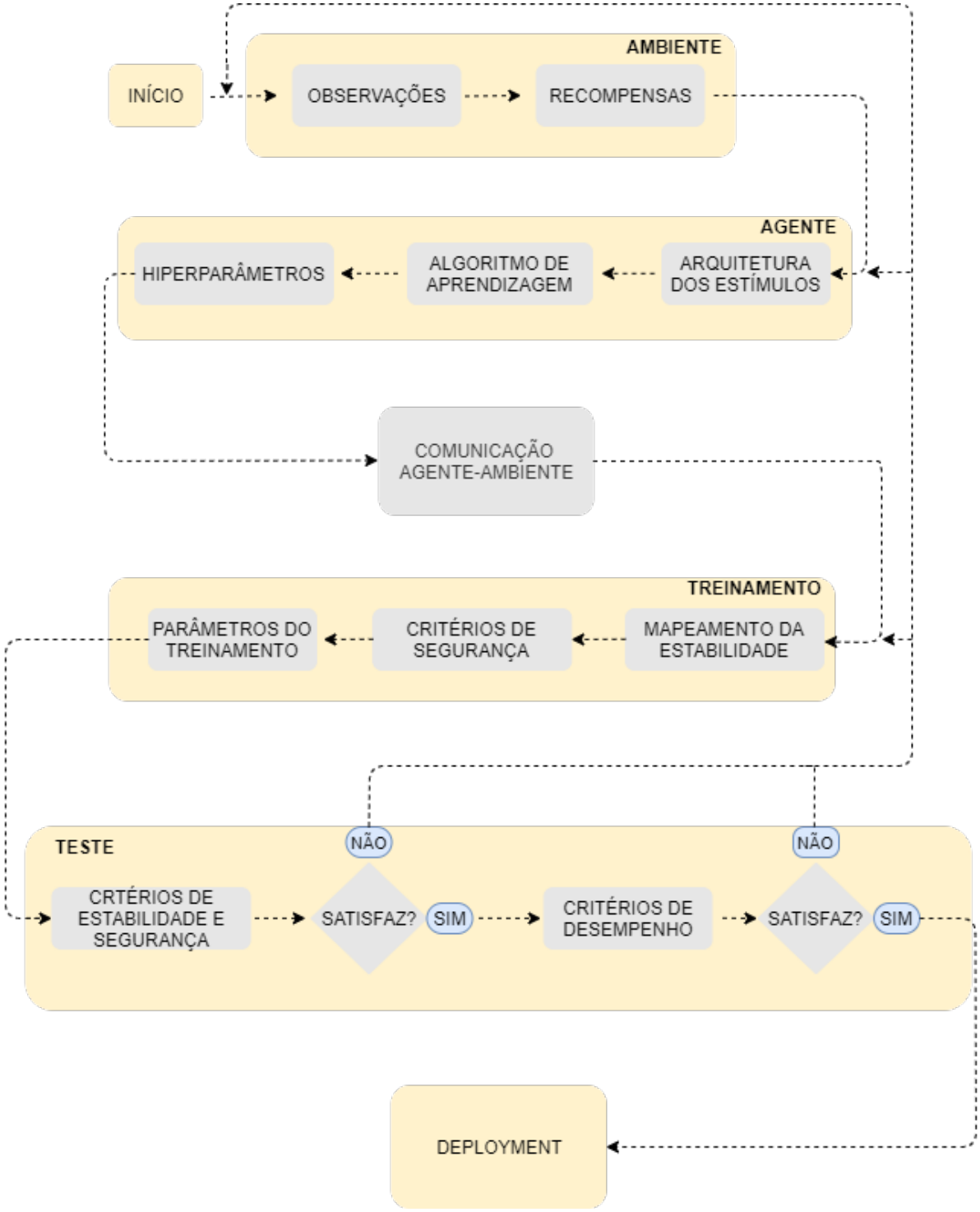
Alguns hiperparâmetros são mais importantes que outros, referindo-se ao efeito que causam no desempenho do treinamento da rede. Existem diversas formas de abordar a escolha dos hiperparâmetros mais sensíveis ao problema, no geral isso também é uma perspectiva empírica.

No caso específico do controlador proposto, além dos hiperparâmetros do próprio agente neural de controle, existem aqueles relacionados ao módulo BEL do controlador emocional, a taxa de aprendizado da amígdala (α) e do córtex (β). Geralmente tais hiperparâmetros apresentam um impacto direto na velocidade transitória da resposta do controlador emocional. Neste trabalho, todos esses hiperparâmetros estão apresentados no Apêndice B deste trabalho.

De modo a facilitar a compreensão do desenvolvimento do controlador DBELBIC, a Figura 47 apresenta o fluxograma das principais etapas da construção deste controlador.

[24] Quando os dados de treinamento são divididos em pequenos lotes, cada lote recebe o nome de mini-batch.

Figura 47 – Etapas da concepção e construção do controlador DBELBIC.



Fonte: próprio autor.

De forma inicial, caso o modelo do ambiente seja conhecido, determinam-se as características do ambiente dinâmico em termos das funções de observação e recompensa, ou seja, quais atributos e variáveis o agente receberá como informação relevante para a tomada de decisão. Além disso, deve-se atentar para o fato de que tais funções são norteadas pela

limitação da disponibilidade dos sensores de monitoramento, caso o objetivo final seja uma aplicação embarcada. Por outro lado, uma vez desconhecido o modelo do ambiente, busca-se primeiramente realizar sua identificação adequada.

Na etapa de desenvolvimento do agente, a escolha da arquitetura dos estímulos é a primeira tarefa a ser concluída, devendo-se optar entre as arquiteturas propostas neste trabalho, direta ou indireta. Após isso, deve-se escolher qual o tipo de algoritmo será utilizado para o aprendizado do agente, que no âmbito desse trabalho, concentra-se nas implementações disponibilizadas pela *OpenAI Baselines*. Por fim, atenta-se aos hiperparâmetros do problema, etapa na qual concentra-se a maior sensibilidade do projeto, uma vez que o agente é formado basicamente por DNNs, contendo uma quantidade razoável de hiperparâmetros que podem afetar de forma significativa a eficácia do controlador.

Uma vez concluída a etapa de dimensionamento do ambiente e do respectivo agente, faz-se necessário estabelecer a forma de comunicação entre ambos. De modo a facilitar a comunicação do agente junto a um ambiente, referindo-se às situações em que ambos estão em plataformas distintas, como é o caso do agente desenvolvido em *Python* e o ambiente no *Simulink*®, optou-se por adaptar a biblioteca padrão da *OpenAI Gym* (quando agente e o ambiente estão em *Python*). Nesse caso, um ambiente adicional é formulado com os moldes padrões da própria biblioteca, diferenciando-se principalmente na forma da troca das informações, que nesse caso pode ser do tipo serial ou TCP/IP, a depender da aplicação final.

A escolha de utilizar a biblioteca *OpenAI Gym* modificada se deu principalmente pela facilidade de implementação, treinamento e *deployment* do agente, uma vez que os algoritmos de aprendizado do agente estão formulados com base nessa própria biblioteca. O código fonte do ambiente modificado a partir do padrão *OpenAI Gym* encontra-se disponível no Apêndice A deste trabalho.

No que diz respeito ao treinamento do sistema, deve-se atentar para alguns aspectos específicos. Primeiramente, utiliza-se um mapeamento prévio de estabilidade com o intuito de simplificar a busca da arquitetura dos sinais de estímulos (arquitetura indireta desconhecida). Em seguida, faz-se necessário impor alguns critérios de segurança ao agente de controle. Uma vez que o espaço de ações de um agente treinado pode apresentar ações acima dos limites físicos do ambiente, deve-se buscar limitá-las dentro de tais limites. Por fim, atenta-se para os parâmetros e hiperparâmetros de treinamento do processo. Nesse caso, é necessário definir uma série de variáveis que podem afetar o resultado final do agente, portanto, tal etapa é uma das mais sensíveis do processo de comissionamento do controlador.

A etapa de teste é constituída por simulações, as quais buscam atestar a eficácia do agente treinado perante o ambiente dinâmico, mais precisamente no que refere-se à produção adequada dos estímulos para o módulo BEL do controlador proposto. Caso os critérios de estabilidade ou os critérios de desempenho não sejam atendidos, ambos definidos previamente, pode ser necessário realizar algumas modificações no modelo e retreiná-lo. Nesta situação, pode-se optar

por modificar alguns hiperparâmetros da etapa de treinamento, os mais sensíveis ao problema, de forma a obter um modelo adequado para a proposta. No entanto, pode-se também verificar se as funções de observação ou recompensas são definidas adequadamente e, em algumas situações pode-se modificar o algoritmo de aprendizagem do agente.

Vale ressaltar que alguns resultados provenientes da etapa de teste, bem como a análise do treinamento podem fornecer informações importantes a respeito de qual etapa ou etapas devem ser modificadas.

Finalizadas todas as etapas anteriores e, caso o controlador atenda a todos os critérios estabelecidos pelo projetista, então pode-se realizar o *deployment* no *hardware* desejado. É importante salientar que mesmo após o *deployment*, faz-se necessário realizar novos testes a fim de assegurar o correto funcionamento do controlador em ambiente real, uma vez que as simulações não conseguem reproduzir em toda sua integralidade a dinâmica de um sistema real de fato.

4.7 Considerações finais

O controlador emocional é conhecido por possuir um modelo de funcionamento baseado nas características do processo de tomadas das decisões que ocorrem no cérebro humano (SHAH-MIRZADI, 2005). Uma das etapas principais no dimensionamento eficiente do controlador emocional se concentra na definição adequada dos sinais de seus estímulos. A proposta de união do controlador emocional às técnicas de DRL possibilitou uma nova abordagem para a definição da arquitetura dos estímulos, denominada neste trabalho de DBELBIC. A partir deste modelo de controlador emocional é possível otimizar seu funcionamento através de diferentes modelos de arquiteturas com o uso de DNNs. O treinamento das DNNs por meio de algoritmos de DRL contribuem para uma modelagem adaptativa de leis de controle não lineares necessárias para os estímulos deste tipo de controlador emocional.

5 RESULTADOS E DISCUSSÕES

Os diversos trabalhos produzidos concernentes às aplicações do aprendizado emocional na engenharia de controle, principalmente nos últimos tempos, atestam para a sua eficácia nesse ramo da engenharia (LOTFI; KHAZAEI; KHAZAEI, 2017).

5.1 Introdução

Apesar de ser uma área relativamente recente, comparada com outros modelos de controladores, o controlador emocional tem apresentado um crescente interesse por parte de diversos pesquisadores (LOTFI; REZAEE, 2018).

No que se refere às ANNs, os últimos avanços na teoria do DL marcou uma nova era em suas aplicações (POUYANFAR et al., 2018). Além disso, diversos avanços em engenharia de controle utilizando as DNNs foram obtidos a partir da junção desses recentes avanços com técnicas de RL (LILLICRAP et al., 2016; MNIH et al., 2015).

A proposta de unir as áreas da aprendizagem emocional com o DRL permitiu desenvolver um novo tipo de controlador emocional (DBELBIC). Desta maneira, é possível apresentar uma nova perspectiva de aplicação do controlador emocional, tornando-o mais abrangente em sua utilização na engenharia de controle.

A partir da realização de diferentes experimentos na área da engenharia de controle, é possível realizar um comparativo desta proposta com outras técnicas, obtendo assim uma perspectiva de sua eficácia como controlador proposto.

5.2 Problemas de rastreamento

Os problemas que envolvem o rastreamento (seguimento de referência) são situações comuns na engenharia de controle. Nesses casos, o controlador tem como objetivo principal garantir o seguimento da referência constante dos sinais padrões[1], proporcionando um erro nulo em regime permanente. Além disso, espera-se que o sistema rejeite assintoticamente as pertubações ao sistema (OGATA, 2003; DORF; BISHOP, 2018).

Neste trabalho, referindo-se ao problema de rastreamento, optou-se por utilizar duas situações problemas encontradas no trabalho de (SHAHMIRZADI, 2005).

O primeiro caso se refere a um sistema simplificado de submarino, o outro é um braço robótico com um grau de liberdade. Dessa maneira, propõe-se realizar um comparativo mais preciso do DBELBIC nessas aplicações junto a outros controladores.

[1] Degrau, rampa, parábola e senoide.

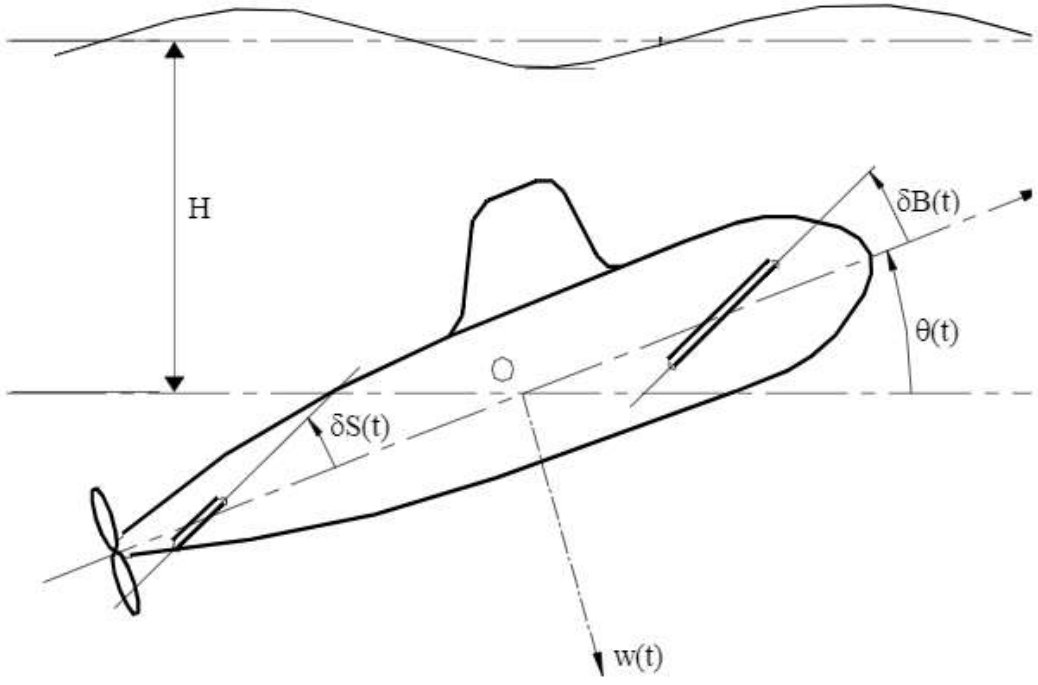
5.2.1 Sistema de submarino

O modelo matemático que descreve a dinâmica de submarinos em águas profundas geralmente é feito a partir de um conjunto de equações não lineares com seis graus de liberdade, baseando-se nas equações de (KIRCHHOFF, 1876). Além desse modelo, outros modelos mais simplificados e " amigáveis" foram desenvolvidos para o estudo da dinâmica de submarinos em geral (GUELER, 1989; TOLLIVER, 1996; MANDZUKA, 1998).

No presente trabalho, o modelo de submarino utilizado é uma versão simplificada, utilizada por (SHAHMIRZADI, 2005) em seus trabalhos. Tal modelo é um sistema SISO[2] linear, o qual é inerentemente instável em malha aberta. Uma vez que uma perspectiva importante da proposta de controlador neste trabalho é realizar um comparativo com o BELBIC, tal situação é justificada.

A Figura 48 apresenta um movimento vertical linear de um submarino e as suas principais coordenadas.

Figura 48 – Modelo de submarino em movimento vertical.



Fonte: adaptado de (MANDZUKA, 1998).

Nesse modelo, H representa a profundidade do submarino em relação à superfície, θ o

[2] Do inglês single input, single output ou entrada única, saída única.

ângulo do submarino em relação a horizontal, w a velocidade de empuxo, δB e δS as deflexões instantâneas da proa e popa do submarino, respectivamente. O objetivo do controlador é que o sistema deve atingir a profundidade desejada debaixo d'água.

De forma a obter um modelo prático de um submarino para análise, diversas manipulações e simplificações algébricas são aplicadas em modelos matemáticos dos trabalhos referentes ao tema, citados anteriormente. Neste caso, o sistema tem como entrada um valor específico de empuxo e a saída do sistema é a profundidade em relação a superfície do oceano. Assim sendo, a função de transferência que representa o modelo de submarino pode ser descrita por

$$G(s) = \frac{0.1(s+1)^2}{s(s^2+0.09)} = \frac{0.1s^2+0.2s+0.1}{s^3+0.09s}. \tag{36}$$

Uma vez obtido o modelo dinâmico que representa o sistema simplificado do submarino, faz-se necessário definir o vetor de observação e a função de recompensa do agente de DRL, responsável por produzir os estímulos sensorial (*S*) e emocional (*R*) do controlador DBELBIC.

De modo a representar a dinâmica deste ambiente, descrito em (36), define-se como as variáveis de observação neste ambiente o *erro*, a *integral do erro* e a *derivada do erro*. Essas variáveis são alocadas em um vetor denominado de *vetor observação*, em que cada posição do vetor representa uma variável observada do ambiente.

A Tabela 5 apresenta as características do *vetor observação* do ambiente dinâmico do submarino.

Tabela 5 – Características do vetor observação do ambiente do submarino.

Ordem	**Observação**	**Limite mínimo**	**Limite máximo**
0	Erro	-∞	∞
1	Integral do Erro	-∞	∞
2	Derivada do Erro	-∞	∞

O ambiente de submarino é capaz de comportar ambos os tipos de agentes, discretos ou contínuos. Neste trabalho, optou-se por utilizar agentes contínuos nos problemas, pois geralmente deseja-se uma variação contínua na ação do agente.

A função do incentivo é formulada de acordo com o objetivo desejado como meta do agente. No caso dos problemas em que envolvam o seguimento de referência, geralmente, utiliza-se o erro mínimo para a obtenção de ganho máximo de recompensa.

Neste trabalho, a função de incentivo para o problema de seguimento de referência do ambiente do submarino é definida por

$$I_{sub}(t) = 10(|e| \leq 0.1) - 1(|e| > 0.05) - 100(y(t) \leq 0||y(t) > 20). \quad (37)$$

De forma geral, a formulação em (37) é feita de tal maneira que, a cada instante de tempo, um erro menor que um valor limiar (0.1) produza um valor de recompensa positivo (+10). Em contrapartida, um erro maior que o valor limiar acrescenta um valor negativo para a recompensa (-1), desencorajando o agente. Além disso, um valor adicional (-100) de punição é associado com valor da variável controlada ($y(t)$), tal que esta variável apresente um peso na dinâmica das decisões do agente.

As funções de incentivo utilizadas na metodologia do presente trabalho, caracterizam-se por funções lógicas, ou seja, caso uma condição específica seja satisfeita, o ganho associado é acrescido na composição final da recompensa equivalente. Assim sendo, em (37) as análises $(|e| \leq 0.1), (|e| > 0.05)$ e $(y(t) \leq 0||y(t) > 20)$ são todas funções lógicas, retornando o valor lógico 1 ou 0 (ativando ou não o ganho associado).

5.2.1.1 Treinamento no ambiente do submarino

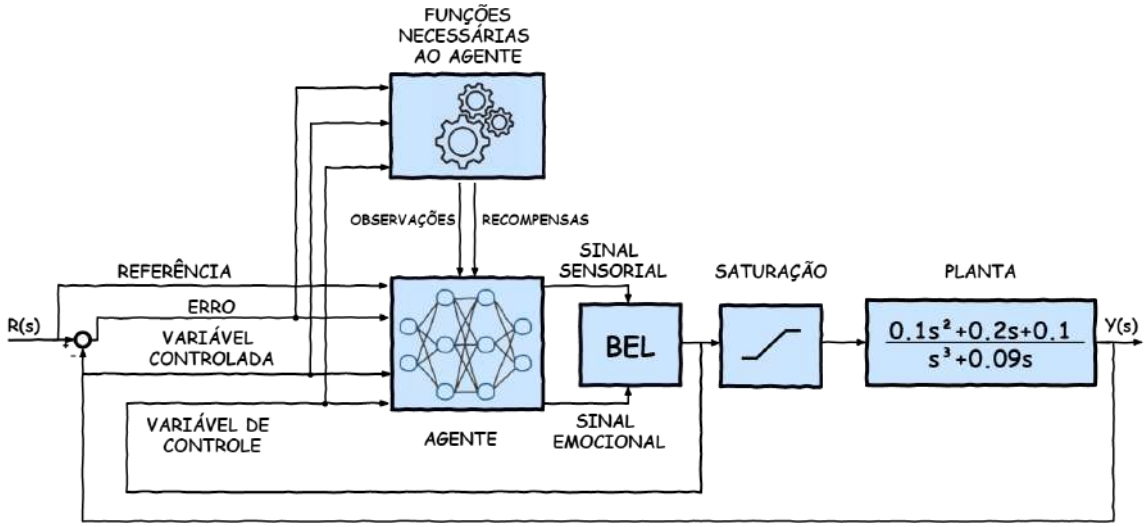
Na etapa de treinamento, é essencial definir o tipo de arquitetura a ser utilizada para compor os estímulos do DBELBIC. No caso do problema em questão, optou-se em primeiro lugar por utilizar uma arquitetura direta, no entanto, observou-se que o sistema treinado não apresentou um desempenho esperado, principalmente no que se refere ao erro em regime permanente.

De fato, a arquitetura direta do agente em problemas de seguimento de referência pode resultar em condições de funcionamento limitadas, uma vez que as ações de controle podem apresentar grandes variações e, nesses casos, as ANNs/DNNs podem não apresentar saídas condizentes.

De maneira a obter o modelo final dos estímulos do DBELBIC para o ambiente do submarino, dividiu-se o treinamento do agente em duas partes. Em primeiro lugar, utilizando-se de um mapeamento prévio da estabilidade, propõe-se obter um modelo mais simplificado da estrutura dos sinais que melhor compõem os estímulos sensorial e emocional. Por fim, utilizando a arquitetura indireta, já de posse das arquiteturas dos estímulos, espera-se obter um refinamento dos ganhos da estrutura dos estímulos.

A Figura 49 apresenta o esquema do controlador DBELBIC utilizado para realizar seu treinamento.

Figura 49 – Esquema do sistema de controle do submarino com DBELBIC utilizado para o treinamento dos estímulos.



Fonte: próprio autor.

A partir da Figura 49, nota-se que o agente de DRL pode fazer uso da variável do *erro*, *referência*, *variável controlada* e da *variável de controle* para compor os sinais de estímulos *S* e *R*, caracterizando a estrutura como arquitetura indireta. Além disso, tais variáveis são utilizadas para gerar as funções auxiliares da observação e de incentivo (funções necessárias ao agente).

No que se refere às características dos agentes de DRL disponíveis na biblioteca *OpenAI Baselines* (BROCKMAN et al., 2016) (Tabela 3), optou-se por realizar o treinamento dos agentes TD3 (FUJIMOTO; HOOF; MEGER, 2018), TRPO (SCHULMAN et al., 2015), PPO (SCHULMAN et al., 2017) e ACKTR (WU et al., 2017b).

A maior motivação para a escolha desses agentes está no fato de que representam aplicações recentes com bom desempenho em diversos problemas em geral. Os parâmetros e hiperparâmetros dos agentes de DRL se encontram descritos no Apêndice B deste trabalho.

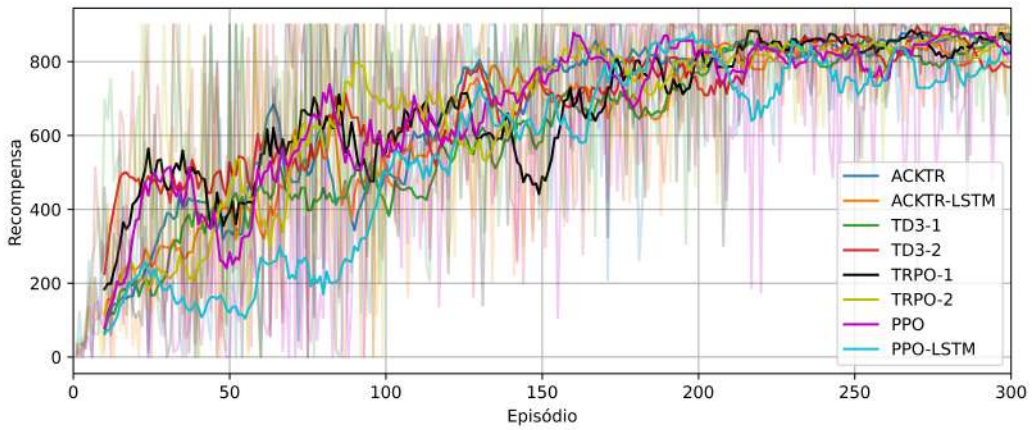
De posse do modelo do ambiente dinâmico do submarino em *Simulink*® e o agente de controle em *Python*, é possível realizar o treinamento do DBELBIC através dos agentes de DRL da biblioteca *OpenAI Baselines* (DHARIWAL et al., 2016) via protocolo de comunicação serial.

A partir de um pré-treinamento inicial, utilizado para o mapeamento da estabilidade com respeito aos estímulos, obtém-se uma melhor noção das variáveis que podem vim a compor tais sinais. Nesse caso, algumas variáveis como a referência (r) e o sinal controlado (y) podem ser descartados, uma vez que influenciam bastante na perda da estabilidade do DBELBIC.

Uma vez finalizado o mapeamento inicial dos estímulos, delimita-se com maior precisão o espaço de ações dos agentes de DRL. Além disso, de forma a diversificar melhor a etapa treinamento, optou-se por fazer uso de duas formas distintas para cada agente de DRL utilizado, seja modificando a arquitetura, identificando-se como LSTM ou não, seja modificando o número de camadas e unidades ocultas, identificando-se como 1 ou 2.

A Figura 50 apresenta o resultado do treinamento, obtido a partir da utilização de diferentes agentes na produção dos estímulos do DBELBIC no ambiente dinâmico do submarino.

Figura 50 – Treinamento do DBELBIC com diferentes agentes no ambiente do submarino.



Fonte: próprio autor.

O treinamento nos problemas do rastreamento da referência é feito aplicando-se diferentes sinais do tipo degrau à entrada do sistema dinâmico e, a partir disso, o agente busca minimizar o erro através de variações nos sinais dos estímulos *S* e *R*. Além disso, o treinamento é estipulado em até *20 s* e, após isso, as condições do ambiente são reiniciadas. Caso a variável controlada ultrapasse um determinado valor, o ambiente também é reiniciado.

No caso do ambiente do submarino, o treinamento demonstrou que os agentes de DRL selecionados foram capazes de obter maiores recompensas à medida que interagiam com este ambiente (Figura 50). Uma vez que o treinamento tinha em média *20 s*, o tempo desta etapa durou em média *1* hora e *40* minutos (cada agente).

No entanto, apesar de bons resultados dos agentes envolvidos, optou-se por selecionar apenas um agente para compor o DBELBIC final. Nesse caso, o agente selecionado foi o TD3-1, justificado por sua performance no treinamento e manipulação amigável. Nesse sentido, após as etapas de mapeamento da estabilidade e consequente treinamento, os sinais de estímulos sensorial e emocional para o ambiente de submarino são formulados da seguinte maneira

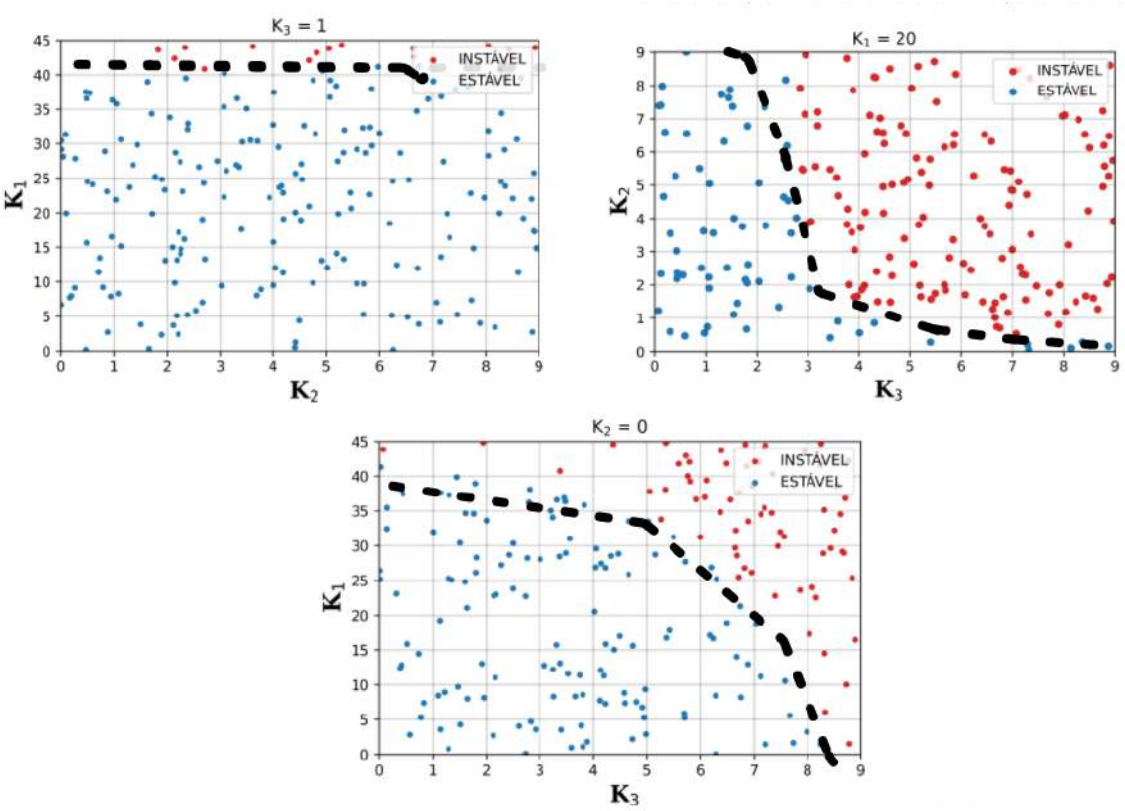
$$S_{sub} = K_1 e, \tag{38}$$

$$R_{sub} = K_2 e + K_3 u, \tag{39}$$

onde K_1, K_2 e K_3 são os ganhos provenientes das DNNs do agente de DRL, *e* é o erro e *u* a variável de controle.

A partir dos ganhos *K* nos estímulos *S* e *R* em (38) e (39), respectivamente, definidos na etapa de treinamento, apresenta-se na Figura 51 um mapeamento de estabilidade em alguns pontos de operação do controlador.

Figura 51 – Mapeamento da estabilidade a partir dos ganhos K_1, K_2 e K_3 no ambiente do submarino.



Fonte: próprio autor.

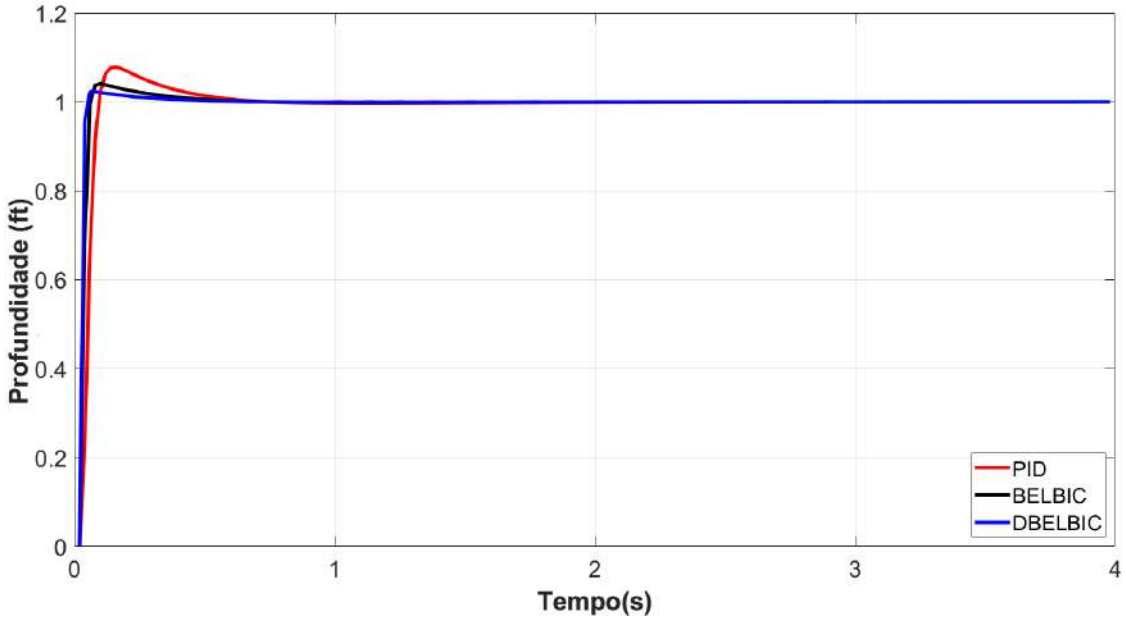
A partir da Figura 51 é possível notar que nestes pontos de operação a relação dos ganhos *K* apresentam distintos comportamentos. Nesse caso, destaca-se que o ganho K_3, associado à variável de controle (*u*) no estímulo R, apresenta uma maior sensibilidade para o controlador, uma vez que uma variação relativamente menor em seus valores causa a instabilidade do sistema.

5.2.1.2 Resultados no ambiente do submarino

Uma vez finalizada a etapa de treinamento da arquitetura dos estímulos, realiza-se uma análise de desempenho do controlador DBELBIC resultante. Para isto, busca-se compará-lo com outros controladores utilizados no mesmo sistema dinâmico, os quais estão presentes no trabalho de (SHAHMIRZADI, 2005).

A Figura 52 apresenta o resultado da resposta do sistema de controle diante de uma entrada tipo degrau unitário.

Figura 52 – Comparativo entre as respostas ao degrau do PID, BELBIC e DBELBIC juntos ao sistema de submarino.



Fonte: próprio autor.

A partir da Figura 52, nota-se que o DBELBIC obteve um melhor desempenho no tempo de subida e *overshoot* relativamente aos outros controladores utilizados. A Tabela 6 apresenta informações referentes a resposta à entrada degrau (Figura 52).

Tabela 6 – Características dinâmicas das respostas do PID, BELBIC e DBELBIC junto ao sistema de submarino.

Controlador	**Sobressinal (%)**	**Tempo de subida (s)**	**Tempo de acomodação (s)**	**Erro estacionário (%)**
PID	9.50	0.26	1.80	0.00
BELBIC	5.15	0.02	0.20	0.00
DBELBIC	2.58	0.02	0.25	0.00

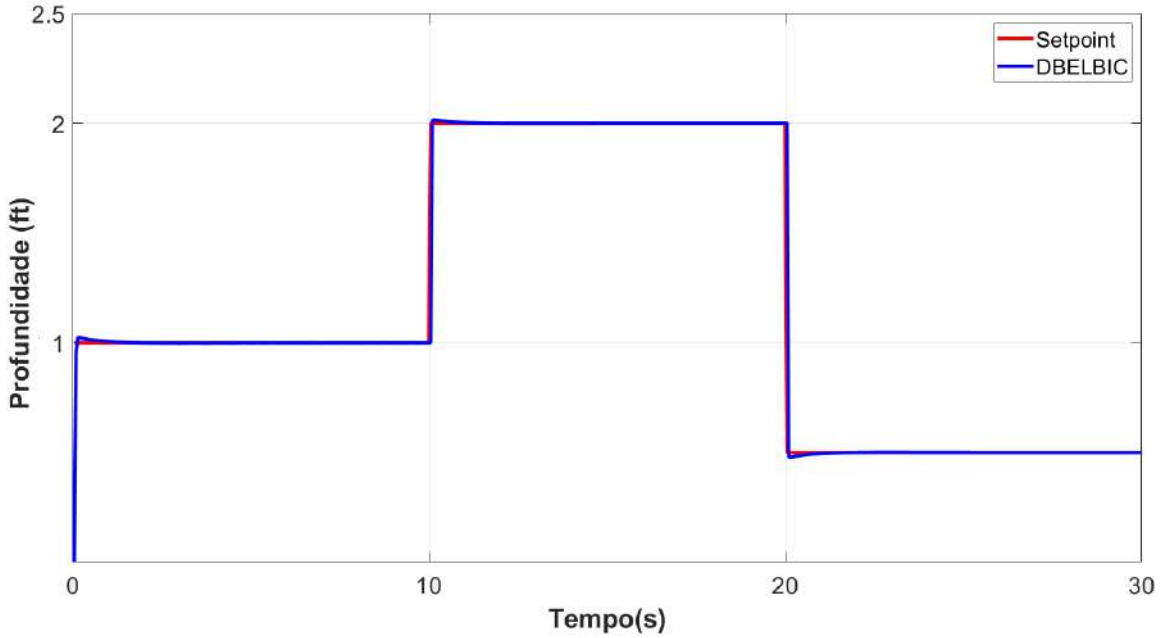
No que se refere aos resultados apresentados na Tabela 6, verifica-se que o controlador DBELBIC apresentou um melhor desempenho dinâmico no rastreamento da referência no sistema do submarino.

Apesar dos controladores PID e BELBIC apresentarem desempenhos muito satisfatórios, o agente treinado para o DBELBIC foi capaz de produzir os estímulos, sensorial e emocional, tal que permitiu que esse controlador superasse o desempenho dos controladores PID e BELBIC tradicional, avaliados junto ao sistema do submarino.

Além disso, uma importante análise para o DBELBIC consiste tanto em verificar o aprendizado da amígdala e do córtex, assim como o comportamento dos sinais dos estímulos produzidos pelo agente de DRL. Assim sendo, aplicam-se entradas do tipo degrau ao sistema de controle do submarino ao longo do tempo e, busca-se observar a variação das grandezas do DBELBIC nesse intervalo.

A Figura 53 apresenta o seguimento de referência do DBELBIC junto ao sistema de submarino mediante a entradas do tipo degrau.

Figura 53 – Resposta do controlador DBELBIC após o treinamento do agente de DRL junto ao sistema de submarino.

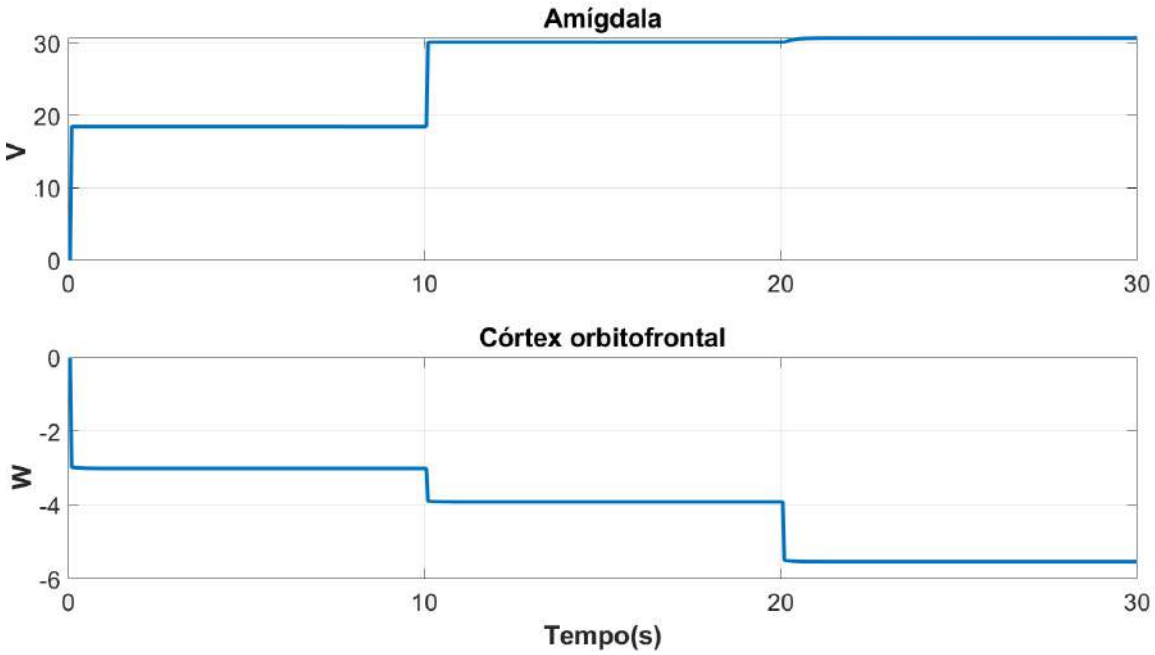


Fonte: próprio autor.

A Figura 53 evidencia a velocidade da resposta produzida pelo DBELBIC. A característica do controlador emocional baseia-se, principalmente, em sua capacidade de rápida adaptação do sistema às variações dinâmicas que ocorram. Tal fato se deve à capacidade de reação dos estímulos, principalmente no que se refere ao estímulo emocional (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004).

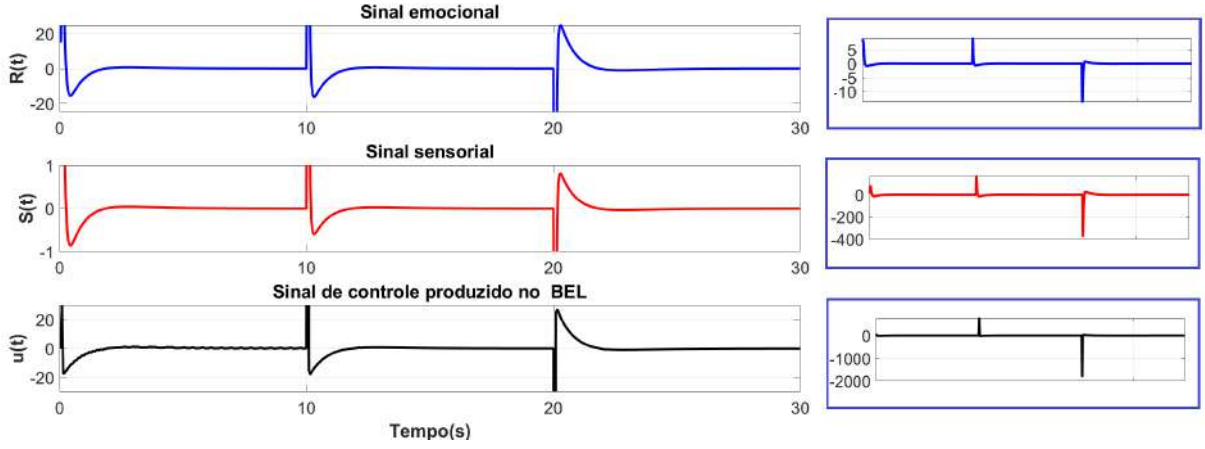
As Figuras 54 e 55 apresentam o comportamento dos módulos de aprendizado da amígdala e córtex do DBELBIC e os sinais produzidos neste controlador, respectivamente.

Figura 54 – Curva de aprendizado do DBELBIC junto ao sistema de submarino.



Fonte: próprio autor.

Figura 55 – Sinais do DBELBIC após o treinamento do agente junto ao sistema de submarino.



Fonte: próprio autor.

A resposta de controle final, associada ao módulo BEL, depende dos estímulos sensorial e emocional recebidos por ele. Como consequência, a velocidade de resposta do controlador, bem como sua adaptação, depende do aprendizado de ambos os módulos, amígdala e córtex.

A partir da Figura 54, nota-se o comportamento monotônico da amígdala, assim como esperado. Apesar de ocorrer um decréscimo no valor de referência e o consequente sinal de controle apresente valor negativo, o aprendizado da amígdala continua a subir. Por outro lado, o comportamento do aprendizado do córtex também condiz com o esperado, uma vez que este age para reprimir o "excesso" da amígdala.

A utilização das ferramentas de DRL disponíveis permitem, cada vez mais, a obtenção leis de controle eficazes à medida em que se utilizam de DNNs, treinadas de forma cada vez mais eficientes e, extraindo melhor as características intrínsecas do sistema por meio da observação dos dados disponíveis (ARULKUMARAN et al., 2017; NAJAFABADI et al., 2015).

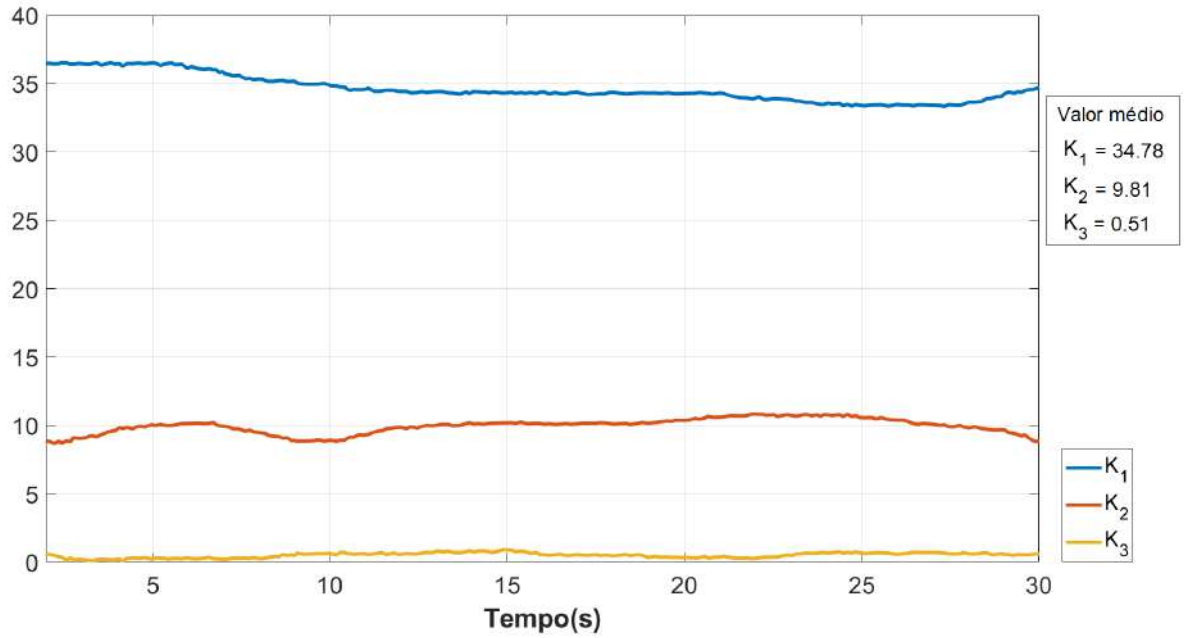
No caso da Figura 55 é possível notar a característica dos estímulos produzidos pelo agente de DRL e o consequente sinal de controle final, produzido pelo módulo BEL. Neste caso é possível notar um impulso nos sinais mediante a mudança no valor de referência (rápida percepção emocional), evidenciada no valor final do sinal de controle (u). Tal fato pode demonstrar o motivo pelo qual a arquitetura direta não foi eficaz neste problema de seguimento de referência, uma vez que uma variação dessa magnitude em um intervalo de tempo muito curto, acaba por dificultar um correto aprendizado do agente em gerar a ação necessária. Por outro lado, uma arquitetura indireta foi capaz de obter um resultado satisfatório no problema de seguimento de referência do sistema de submarino (Figura 53).

Uma vez que a arquitetura indireta se utiliza dos próprios sinais característicos da malha de controle (*erro, variável controlada e variável de controle*) associados a ganhos controlados pela DNN, a complexidade da tarefa da rede neural em otimizar o controlador é simplificada.

Uma vez que o aprendizado da amígdala e do córtex obedecem as formulações descritas em (26) e (27), respectivamente, seus valores dependem de como são arquitetados ambos os estímulos (*S* e *R*). No caso do presente trabalho, um agente de DRL é o responsável em produzir esses valores, os quais, a depender do ambiente dinâmico e da arquitetura (direta ou indireta), pode fornecer ganhos significativos de desempenho a este controlador.

A Figura 56 apresenta o comportamento dos ganhos nos estímulos *S* (38) e *R* (39) em relação ao comportamento do controlador DBELBIC descrito na Figura 53.

Figura 56 – Comportamento dos ganhos nos estímulos do controlador DBELBIC a partir de entradas do tipo degrau unitário.



Fonte: próprio autor.

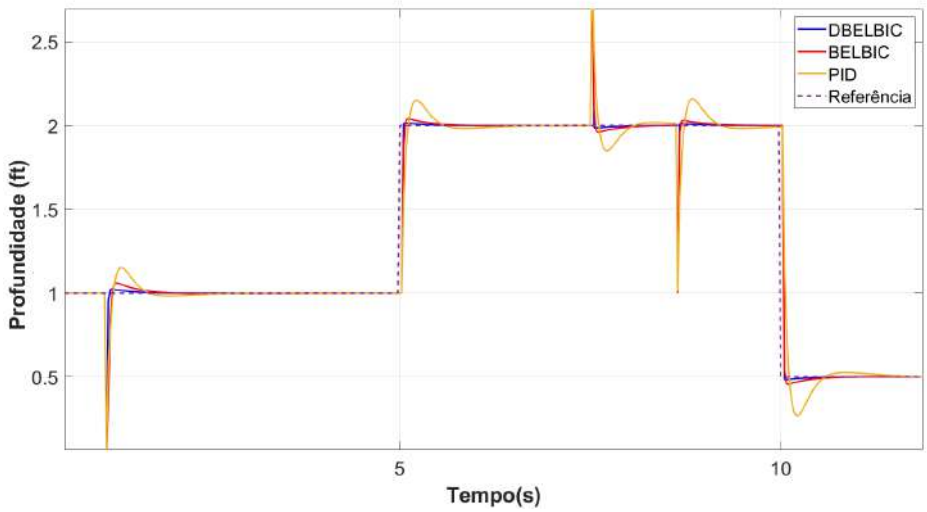
De forma geral, o comportamento dos ganhos K_1, K_2 e K_3 (Figura 56) no ambiente dinâmico do submarino, referindo-se ao funcionamento descrito na Figura 53, apresentou um comportamento estável. Apesar do sistema apresentar variações no sinal da referência e, além disso, os ganhos oscilarem um pouco ao longo do tempo, os respectivos valores mantiveram-se relativamente dentro de uma faixa. No caso do sistema exigir uma maior confiança a respeito da variação dos ganhos K, pode-se optar por limitar mais ainda a faixa de atuação dos agentes de DRL e, dependendo do caso, utilizar os valores médios dos ganhos.

Assim como é importante verificar o desempenho da resposta ao degrau de um sistema de controle, a análise de sua capacidade em rejeitar distúrbios inerentes à malha de controle é essencial. De forma geral, os sistemas dinâmicos estão sujeitos a diversos tipos de pertubações, no entanto, é desejável que tais fenômenos sejam minimizados ou até mesmo completamente anulados pelo controlador após um certo período transitório.

De forma a avaliar o comportamento do controlador DBELBIC diante uma pertubação, aplicam-se sinais do tipo degrau transitório na saída da planta ao longo do tempo.

A Figura 57 apresenta o comportamento dos controladores PID, BELBIC e DBELBIC perante distúrbios no sistema de submarino.

Figura 57 – Controladores DBELBIC, BELBIC e PID junto ao sistema de submarino na presença de pertubações.



Fonte: próprio autor.

A Figura 57 demonstra que todos os controladores obtiveram resultados satisfatórios em reprimir o distúrbio aplicado ao sistema de submarino. No caso do DBELBIC, nota-se que este apresentou um pouco de melhora na resposta comparando-o com BELBIC original, porém, ambos apresentam uma atuação muito semelhante diante da presença da pertubação.

De modo a avaliar a performance dos controladores sob condições de ruídos e distúrbios presentes, utilizam-se os critérios de avaliação da performance do erro *ISE*[3], *IAE*[4] e *ITAE*[5]. A Tabela 7 apresenta um comparativo dos índices da performance do erro na operação dos controladores PID, BELBIC e DBELBIC, descrita na Figura 57.

Tabela 7 – Índices de desempenho do PID, BELBIC e DBELBIC junto ao sistema do submarino associado a ruídos e distúrbios.

Controlador	ISE	IAE	ITAE
PID	5.52	17.54	$4.84\ 10^3$
BELBIC	3.51	6.93	$1.53\ 10^3$
DBELBIC	3.01	4.16	983.58

[3] Do inglês Integral of square error ou Integral do erro quadrático.
[4] Do inglês Integral of absolute error ou Integral do erro absoluto.
[5] Do inglês Integral of time multiplied by the absolute value of error ou Integral de tempo multiplicado pelo valor absoluto do erro.

A partir da Tabela 7, é possível perceber que os controladores apresentaram índices próximos de *ISE*, *IAE* e *ITAE*. Neste caso, destaca-se que o controlador DBELBIC obteve os menores índices dos erros avaliados.

Além das avaliações de desempenho do sistema de controle, mencionadas anteriormente, faz-se de grande relevância prática verificar a robustez do controlador frente às variações paramétricas da planta. Uma vez que os sistemas dinâmicos estão geralmente em constantes variações paramétricas, influenciadas por condições externas ou internas, é importante que um controlador seja capaz de manter um bom desempenho em meio a essas variações na planta.

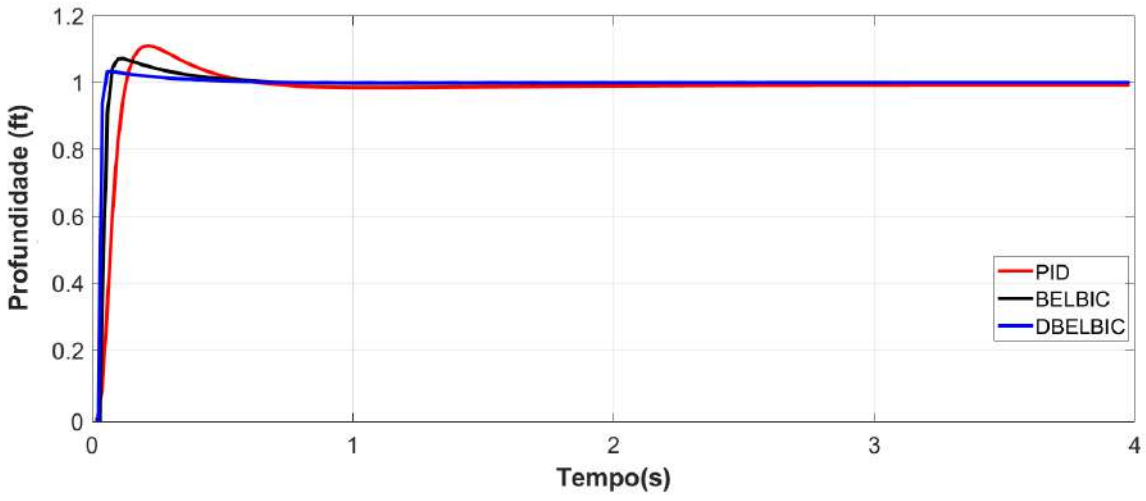
A partir do modelo do sistema de submarino, descrito em (36), realiza-se um variação paramétrica de forma a tornar esse sistema um pouco diferente. A partir de mudanças paramétricas em (36) de acordo com (SHAHMIRZADI, 2005), a nova equação do modelo do sistema de submarino pode ser descrita por

$$G(s) = \frac{0.1s^2 + 0.1s + 0.2}{1.1s^3 + 0.12s + 1}. \tag{40}$$

Assim como na primeira avaliação realizada, aplica-se um sinal do tipo degrau ao sistema de submarino e utilizando os mesmos controladores, PID, BELBIC e DBELBIC, observam-se suas respostas diante da nova condição paramétrica do sistema de submarino.

A Figura 58 apresenta a resposta dos controladores PID, BELBIC e DBELBIC mediante uma entrada tipo degrau do sitema descrito em (40).

Figura 58 – Comparativo entre as respostas ao degrau do PID, BELBIC e DBELBIC juntos ao sistema de submarino modificado.



Fonte: próprio autor.

De acordo com a Figura 58 é possível observar que todos os controladores avaliados apresentarem um pequeno decréscimo de desempenho no seguimento da referência.

Uma melhor comparação entre os diferentes controladores é apresentada na Tabela 8. A Tabela apresenta os valores das respostas dinâmicas em detalhes, provenientes dos controladores, mediante a aplicação do sinal degrau ao sistema de submarino modificado.

Tabela 8 – Características dinâmicas de robustez das respostas do PID, BELBIC e DBELBIC junto ao sistema de submarino modificado.

Controlador	**Sobressinal (%)**	**Tempo de subida (s)**	**Tempo de acomodação (s)**	**Erro estacionário (%)**
PID	11.26	0.27	4.10	-6.63
BELBIC	5.15	0.02	0.93	-1.65
DBELBIC	3.02	0.02	0.82	-1.80

A partir da Tabela 8, nota-se que em relação ao sistema original o BELBIC não apresentou mudança no valor de sobressinal, permanecendo com o mesmo valor anterior ‘à modificação do sistema (5.15%). Por outro lado, o DBELBIC apresentou um aumento no sobressinal (3.02%) assim como o controlador PID (11.26%). No entanto, apesar de tal situação, o DBELBIC ainda demonstrou praticamente os melhores resultados em todas as características observadas. Todas essas análises de desempenho do controlador DBELBIC, apresentadas anteriormente, são de fundamental importância para sua concepção e aplicação.

Apesar dos resultados satisfatórios de desempenho do DBELBIC, obtidos a partir das análises anteriores, tal condição de funcionamento do submarino é restrita a uma análise teórica. De fato, um submarino real não apresenta a velocidade de resposta dinâmica verificada nas simulações avaliadas anteriormente.

Por fim, busca-se abordar a análise da estabilidade do DBELBIC na malha de controle. Tal situação ocorre através da aproximação do sistema de controle total por meio de uma função de transferência equivalente, como mencionado anteriormente na seção 4.5.3 deste trabalho.

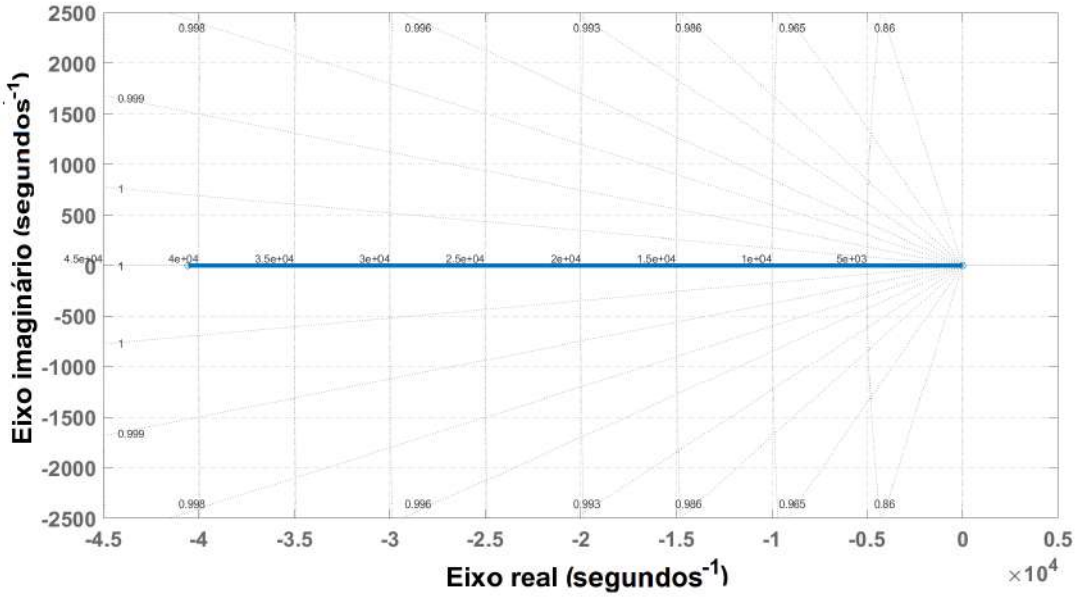
A partir das ferramentas de identificação de sistemas dinâmicos (*identification toolbox*), disponíveis no *software MATLAB®*, é possível obter uma função de transferência aproximada do sistema. Nesse caso, a função de transferência mais relevante obtida para o sistema de controle em questão, envolvendo o DBELBIC e a planta do submarino (36) é descrita da seguinte maneira

$$G_{sub}(s) = \frac{0.002s^4 + 8.55s^3 + 1.52s^2 + 0.3583s}{s^4 + 9.026s^3 + 1.56s^2 + 0.3583s}. \tag{41}$$

Existem alguns métodos para avaliar a estabilidade dos sistemas dinâmicos, porém neste trabalho, os métodos são restritos ao descritos anteriormente na seção 4.5. Nesse sentido, primeiramente, obtém-se o LR do sistema por meio das ferramentas disponíveis no *software MATLAB®*.

A Figura 59 apresenta a região do LR do sistema descrito em (41).

Figura 59 – LR do sistema aproximado pelo DBELBIC junto ao submarino.



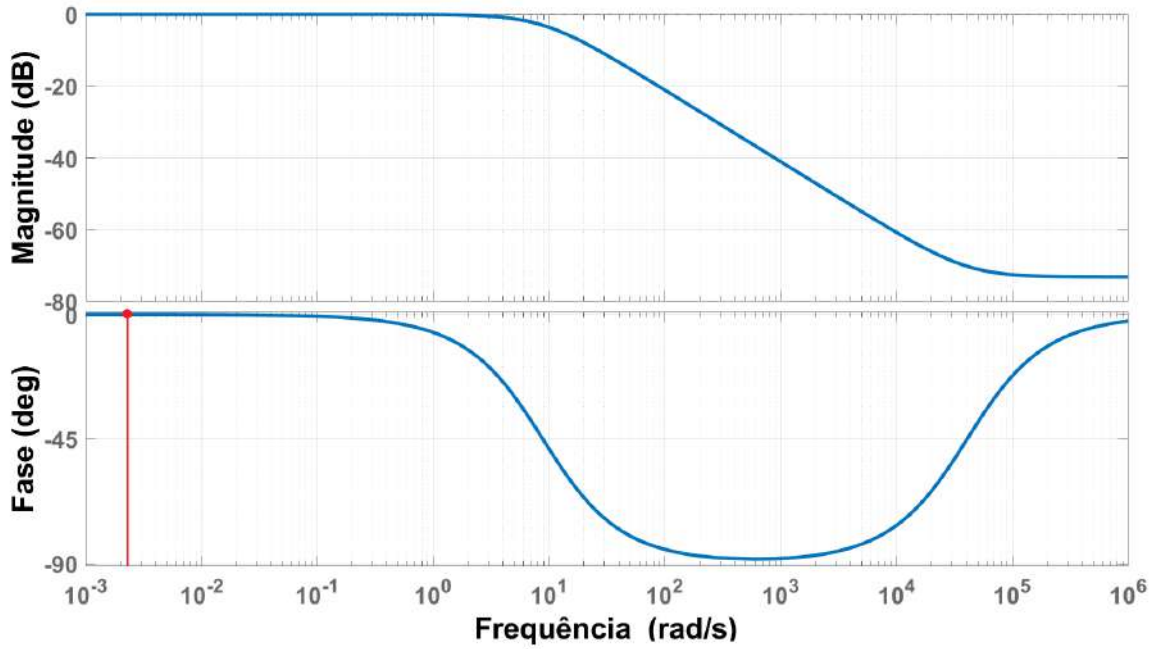
Fonte: próprio autor.

A partir da observação da Figura 59, é possível notar que a localização do LR em (41) está apenas no eixo real negativo do plano, o que denota a não existência da parcela oscilatória do sistema. De fato, tal aproximação de função de transferência demonstra que o sistema é estável para qualquer que seja a posição das raízes do polinômio característico em (41).

Além da análise da estabilidade do sistema pelo LR, optou-se também por verificar o comportamento do *Diagrama de Bode* do sistema. Tal situação pode ser justificada para reforçar a análise da estabilidade do sistema, uma vez que somente o LR pode não contemplar todas as informações necessárias para uma avaliação conclusiva da estabilidade. Assim sendo, obteve-se o *Diagrama de Bode* da equação 41 com o auxílio das ferramentas disponíveis do *MATLAB*®.

A Figura 60 apresenta o *Diagrama de Bode* da função de transferência do sistema envolvendo o DBELBIC e o sistema de submarino (Figura 49).

Figura 60 – Diagrama de Bode do sistema aproximado pelo DBELBIC junto ao submarino.



Fonte: próprio autor.

De fato, a partir da Figura 60 demonstra-se que o sistema como um todo apresenta uma estabilidade robusta, uma vez que ambas as margens de estabilidade (fase e magnitude) são infinitas.

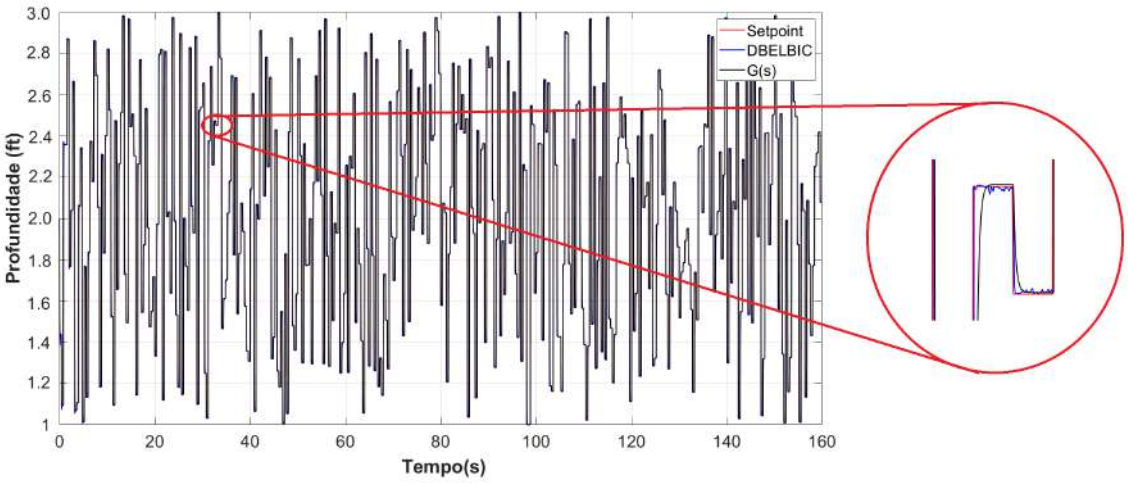
5.2.1.3 A taxa de aprendizado do DBELBIC

As avaliações concernentes ao desempenho dinâmico do controlador DBELBIC, realizadas em seções anteriores neste trabalho, apresentaram resultados satisfatórios nos parâmetros avaliados. Todavia, notou-se em diversos experimentos que, dependendo das mudanças ocorridas na referência (rastreamento de referência), bem como na estrutura dos estímulos, o sinal deste controlador pode apresentar diversos ruídos.

De forma a ilustrar tal observação, um sinal aleatório foi fornecido como valor de referência ao controlador DBELBIC (Figura 49), bem como para a respectiva função de transferência aproximada (equação 41) do sistema.

A Figura 61 apresenta a resposta do sistema de controle representado na Figura 49 e da função de transferência aproximada do sistema de controle definida em (41).

Figura 61 – Simulação a partir de uma entrada aleatória com o DBELBIC junto ao sistema de submarino e de $G_{sub}(s)$.



Fonte: próprio autor.

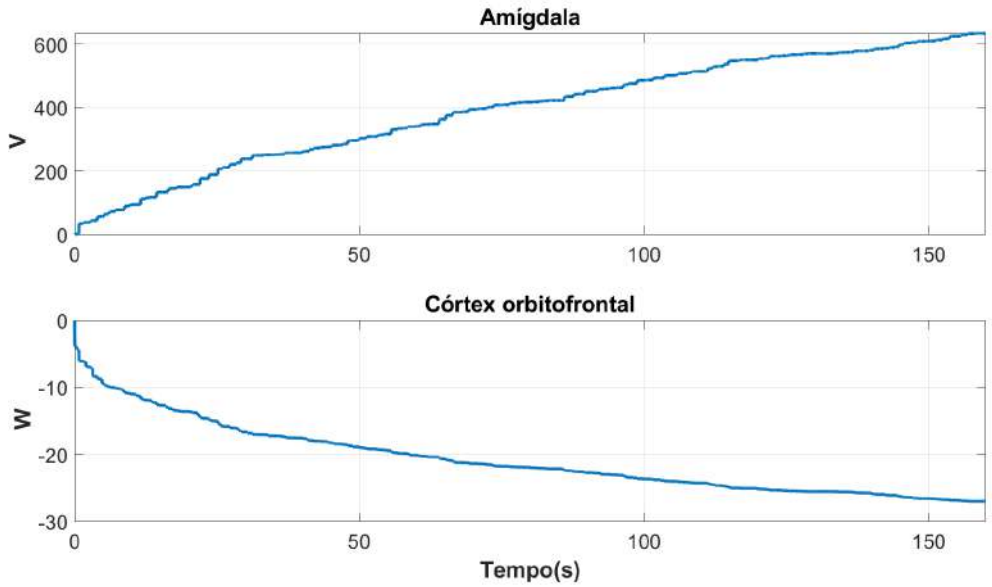
A partir da Figura 61 é possível observar o comportamento oscilatório na saída da planta, referindo-se a utilização do controlador DBELBIC. Por outro lado, a função de transferência aproximada $G_{sub}(s)$ não apresenta tal situação.

A razão desse fenômeno se dá pela forma como acontecem os aprendizados da amígdala e do córtex no módulo BEL do controlador emocional. A cada alteração no valor de referência, ambos amígdala e córtex, precisam aprender a lidar com essa mudança, alterando desta maneira o valor do aprendizado. Caso o controlador atue em um tempo relativamente longo, o valor do aprendizado pode ser muito extenso e, a partir disso, pode-se resultar em sinais de controle demasiadamente grandes.

Uma vez que o sinal de referência fornecido à entrada do sistema de controle apresenta um comportamento aleatório em um intervalo relativamente curto ao longo do tempo e, além disso, associado às características construtivas dos estímulos *S* e *R* em (38) e (39), observou-se que a amígdala e o córtex apresentaram uma variação constante. Esse comportamento busca tornar o sinal final do controlador adaptado à dinâmica dos sinais envolvidos na construção de *S* e *R*. Além disso, a critério de análise, considera-se neste exemplo um valor de aprendizado da amígdala ($\alpha = 0.5$) e do córtex ($\beta = 0.5$), valores relativamente maiores do que comumente utilizados.

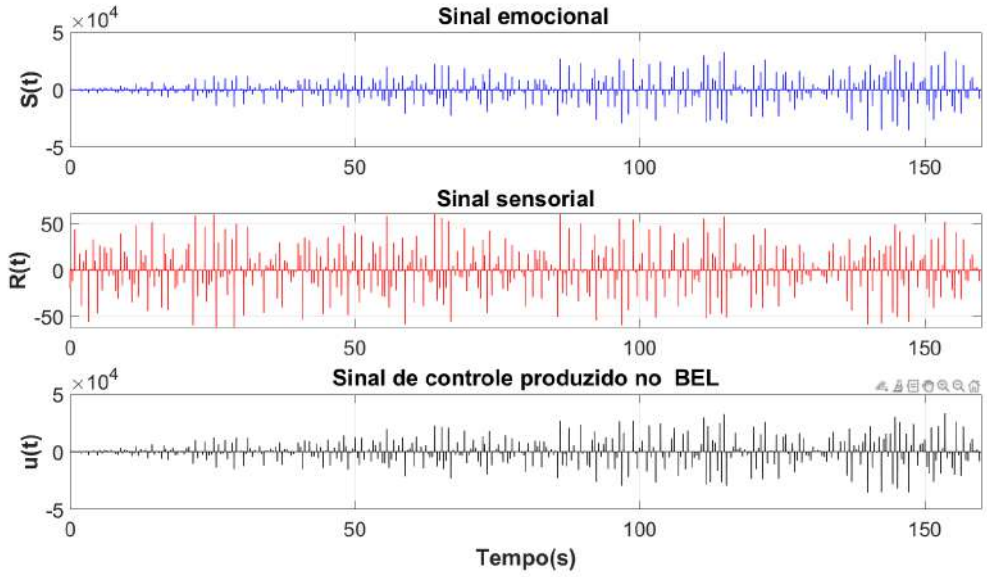
A Figura 62 apresenta, respectivamente, o comportamento dos aprendizados da amígdala e do córtex na simulação anterior (Figura 61). Além disso, a Figura 63 demonstra os sinais resultantes em cada módulo do controlador.

Figura 62 – Curva de aprendizado do DBELBIC junto ao sistema de submarino a partir de uma entrada aleatória.



Fonte: próprio autor.

Figura 63 – Sinais do DBELBIC junto ao sistema de submarino a partir de uma entrada aleatória.



Fonte: próprio autor.

A partir da observação das Figuras 62 e 63, é possível notar uma tendência de crescimento contínuo nos valores de aprendizados (V e W) à medida que o valor da referência varia no tempo e, consequentemente, em valores finais de controle mais elevados.

De forma a minimizar esse efeito provocado pelo aprendizado do controlador emocional, propõe-se neste trabalho uma modificação no valor de ambos os aprendizados do DBELBIC. Essa mudança tem o propósito de provocar uma desaceleração dos valores de aprendizados. As equações dos aprendizados podem ser então redefinidas como

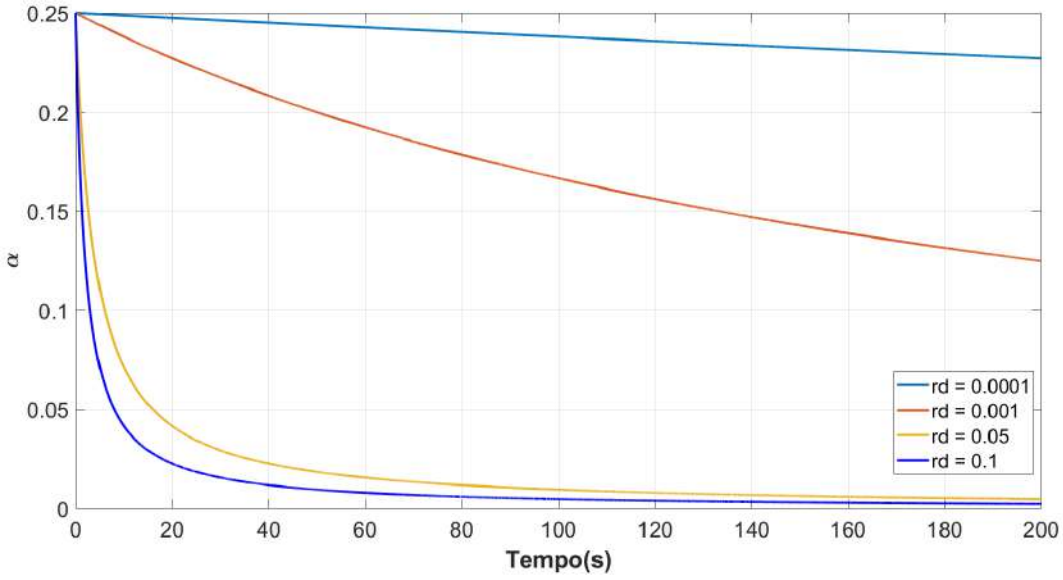
$$\alpha = \frac{1}{(1 + r_d t_s)} \alpha_0, \tag{42}$$

$$\beta = \frac{1}{(1 + r_d t_s)} \beta_0, \tag{43}$$

onde α_0 e β_0 são as taxas inciais de aprendizagem da amígdala e do córtex, respectivamente, r_d a taxa de decaimento e t_s o *timestep* (passo de tempo). Nesse caso, diferentemente dos aprendizados tradicionais (α_0 e β_0) constituírem-se de valores fixos durante todo o tempo de funcionamento do controlador, os valores de α e β apresentam um comportamento de decaimento em seus valores. Caso haja a necessidade de um tempo demasiadamente grande de funcionamento, pode-se também adicionar uma constante a essas relações.

A Figura 64 apresenta diferentes curvas[6] para os aprendizados *V* e *W* de acordo com (42) e (43).

Figura 64 – Curva de aprendizado do DBELBIC junto ao sistema de submarino a partir de uma entrada aleatória.

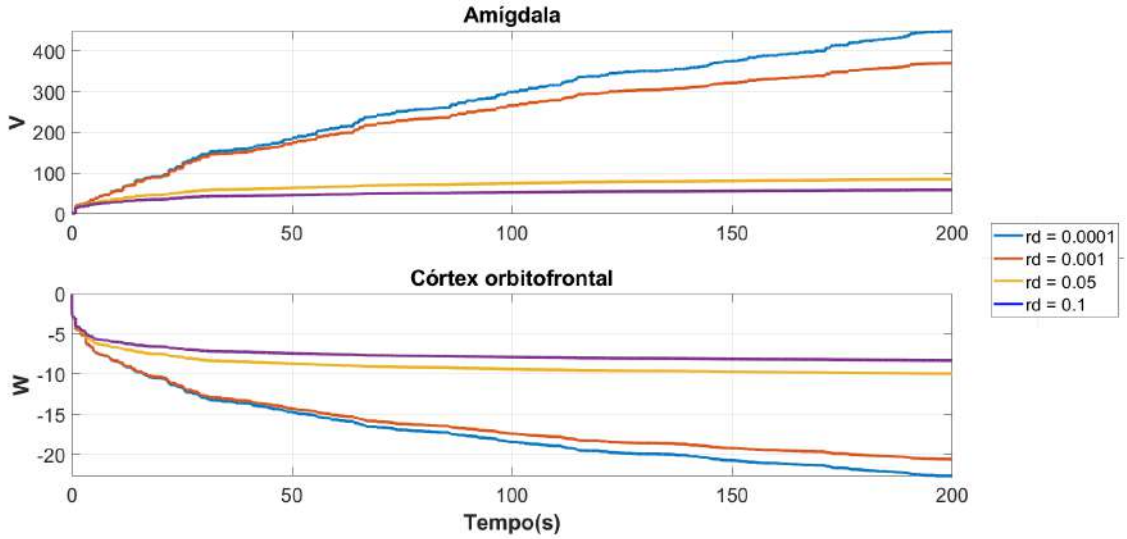


Fonte: próprio autor.

As Figura 65 e 66 apresentam o comportamento dos novos modelos de aprendizados e os consequentes sinais resultantes de cada módulo do DBELBIC, respectivamente ($r_d = 0.05$).

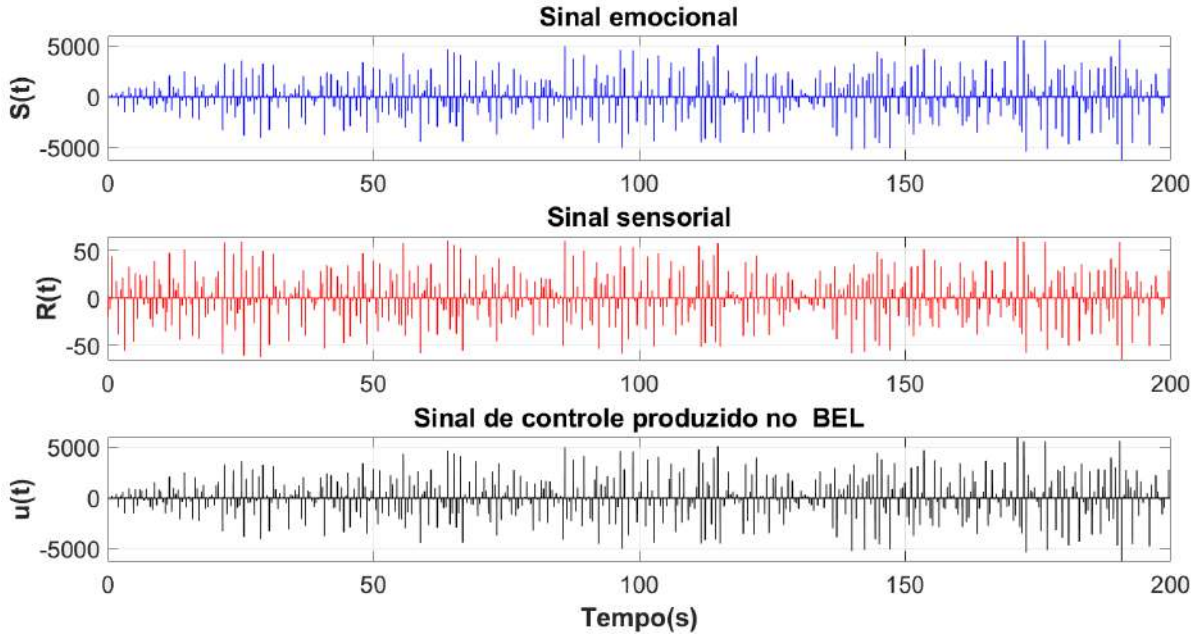
[6] De maneira a facilitar o entendimento faz-se $V = W$.

Figura 65 – Curva de aprendizado do DBELBIC com aprendizado amenizado junto ao sistema de submarino a partir de uma entrada aleatória.



Fonte: próprio autor.

Figura 66 – Sinais do DBELBIC com aprendizado amenizado ($r_d = 0.05$) junto ao sistema de submarino a partir de uma entrada aleatória.



Fonte: próprio autor.

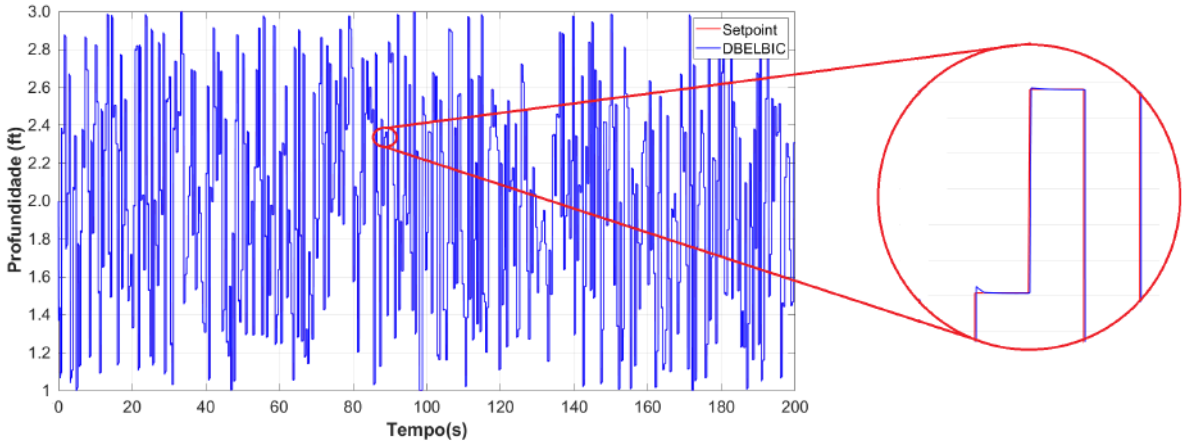
A seleção dos parâmetros em (42) e (43) depende do tempo de atuação e características do controlador. Por outro lado, pode optar-se por utilizar um valor de aprendizado constante, relativamente mais baixo do que a ordem de grandeza comumente utilizada (10^{-1}). Contudo, tal

escolha pode afetar o desempenho do controlador, uma vez que pode vir a retardar sua velocidade inicial de resposta.

A partir da amortização do aprendizado do DBELBIC, simulou-se o controlador da mesma forma como no primeiro caso (α e β contínuos).

A Figura 67 apresenta o comportamento do sistema de controle envolvendo a planta do submarino e o DBELBIC com o aprendizado amortecido ($r_d = 0.05$ e $\alpha_0 = \beta_0 = 0.25$).

Figura 67 – Simulação do DBELBIC com aprendizado amenizado junto ao sistema de submarino a partir de uma entrada aleatória.



Fonte: próprio autor.

A partir da Figura 67, é possível notar que há presença de ruídos na saída da planta, evidenciado na imagem em detalhe.

O uso do controlador DBELBIC no sistema de submarino mostrou-se bastante satisfatório, tanto em termos da arquitetura dos estímulos sensorial e emocional, bem como na velocidade de resposta transitória deste controlador. Todos as simulações e resultados foram obtidos comparando-os com os controladores PID e BELBIC, utilizados em trabalhos relacionados ao tema do controlador emocional.

O sistema do submarino, apesar de apresentar uma boa análise para o controlador proposto, por si só não é suficiente para a obtenção de uma maior "generalidade" desta aplicação. Por esta razão, outros sistemas dinâmicos mais complexos são utilizados para compor a análise do DBELBIC neste trabalho.

5.2.2 Braço robótico com um grau de liberdade

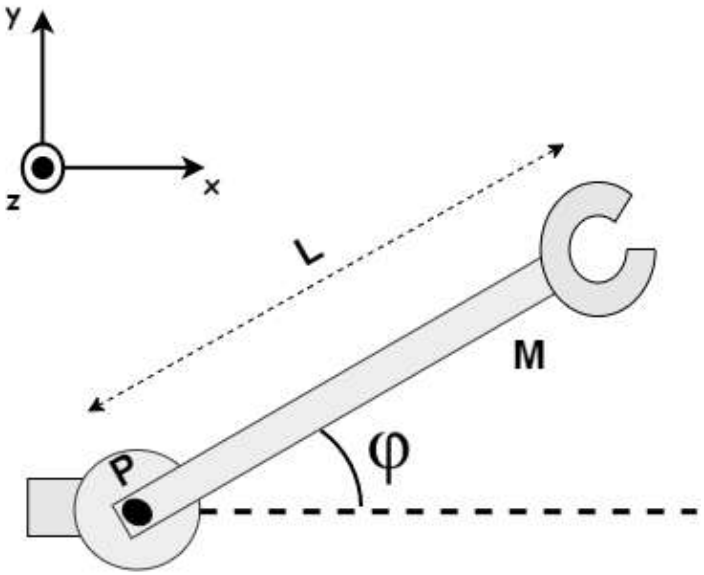
De maneira a avaliar o desempenho do controlador proposto em um ambiente envolvendo dinâmicas não lineares, propõe-se sua utilização em um sistema dinâmico descrito por um braço robótico com um grau de liberdade.

A utilização de manipuladores robóticos no manuseio de objetos é uma área essencial no estudo da robótica. A complexidade do manipulador robótico está diretamente relacionada ao seu grau de liberdade. De forma geral, o grau de liberdade está associado à quantidade de movimentos independentes que o robô pode executar em relação ao eixo das coordenadas (ALCÂNTARA; ALCÂNTARA, 2013).

Neste trabalho, optou-se por utilizar um sistema de braço robótico com um grau de liberdade, uma vez que a dinâmica não linear de tal sistema representa um desafio suficiente à proposta do controlador DBELBIC. Além disso, deseja-se compará-lo com situações relacionadas ao tema do controlador emocional.

A Figura 68 apresenta um sistema de braço robótico com um grau de liberdade e suas coordenadas.

Figura 68 – Braço robótico com 1 grau de liberdade.



Fonte: próprio autor.

O sistema descrito na Figura 68 é formado por um mecanismo de braço robótico com um comprimento L e uma massa M, conectado a um ponto fixo P. O ângulo que o braço robótico faz com a horizontal é φ. Nessa situação, o movimento do braço robótico é realizado apenas no plano x-y. O objetivo do controlador neste caso é aplicar um torque no ponto fixo P do manipulador robótico tal que este atinja um determinado valor angular de referência.

De modo a realizar comparações efetivas do controlador DBELBIC, a equação diferencial do movimento angular, referente ao sistema não linear do braço robótico, baseia-se na utilizada no trabalho de (LOTFI; REZAEE, 2018). O modelo admite como entrada um sinal de torque proveniente de um motor CC. A saída, por outro lado, é a respectiva posição angular (φ). A equação do modelo é descrita por

$$\ddot{\varphi} = -10sin\varphi - 2\dot{\varphi} + u, \tag{44}$$

onde φ é a posição angular do braço robótico e u o sinal de entrada do sistema oriundo de um motor CC.

No sistema descrito em (44), o *vetor observação*, assim como no caso do sistema do submarino, apresenta como grandezas observadas pelo agente de DRL, o *erro*, a *integral do erro* e a *derivada do erro*. Uma vez que se trata de um problema de rastreamento da referência, optou-se por utilizar os mesmos tipos de informações para compor a observação do ambiente dinâmico.

A Tabela 9 apresenta as características do *vetor observação* do ambiente dinâmico do braço robótico com um grau de liberdade.

Tabela 9 – Características do vetor observação do ambiente do braço robótico com um grau de liberdade.

Ordem	**Observação**	**Limite mínimo**	**Limite máximo**
0	Erro	-∞	∞
1	Integral do Erro	-∞	∞
2	Derivada do Erro	-∞	∞

A função de incentivo deste ambiente se assemelha à verificada no ambiente do submarino, uma vez que em ambos os casos tem-se problemas que se referem ao seguimento de referência. O valor do erro em regime permanente é utilizado como principal critério para o recebimento das recompensas do ambiente. Neste caso, a função de incentivo é formulada por

$$I_{arm}(t) = 30(|e| \leq 0.1) - 1(|e| > 0.01) - 100(\varphi \leq 300||\dot{\varphi} > 25). \tag{45}$$

O valor limiar de erro (0.1) é definido para a produção de valores positivos de recompensa (+30). No caso dos valores acima do limiar do erro, recompensas negativas (-1) são acrescidas ao valor da função e, caso o sistema extrapole um valor limite do ângulo (φ) ou velocidade angular ($\dot{\varphi}$), o valor de punição é acrescido (-100) e o ambiente dinâmico é reiniciado.

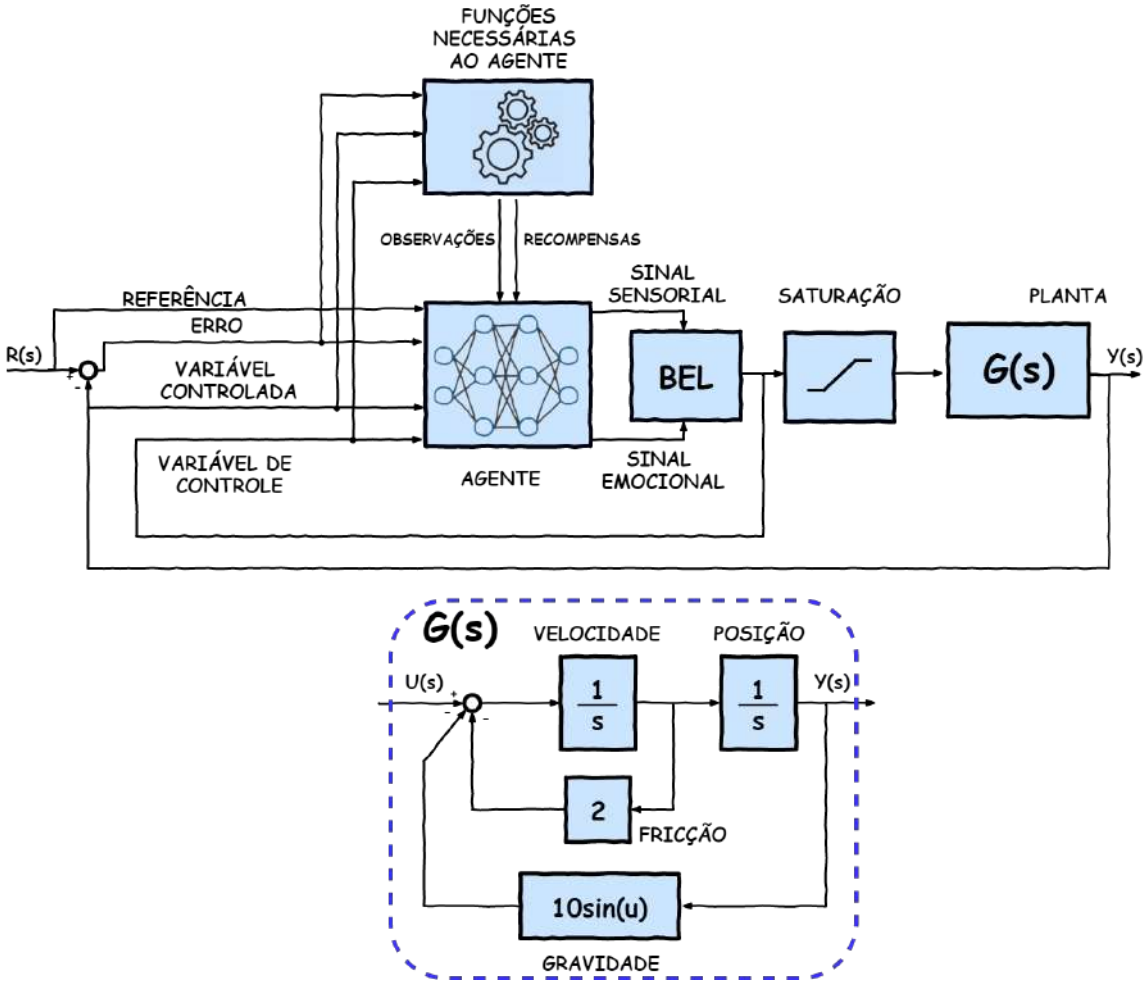
5.2.2.1 Treinamento no ambiento do braço robótico

O processo de treinamento do agente de DRL no sistema do braço robótico é realizado de forma igual ao caso do ambiente do submarino. A primeira etapa constitui em realizar um

mapeamento prévio da estabilidade a partir da seleção dos possíveis sinais candidatos a formarem os estímulos S e R. Em seguida, realiza-se o treinamento dos agentes de DRL selecionados, limitando-se as ações em faixas de operação estáveis para o controlador no respectivo ambiente dinâmico.

A Figura 69 apresenta o esquema do sistema de controle com o DBELBIC, utilizado para realizar seu treinamento no ambiente do braço robótico com um grau de liberdade.

Figura 69 – Sistema de controle do braço robótico com um grau de liberdade junto ao DBELBIC.



Fonte: próprio autor.

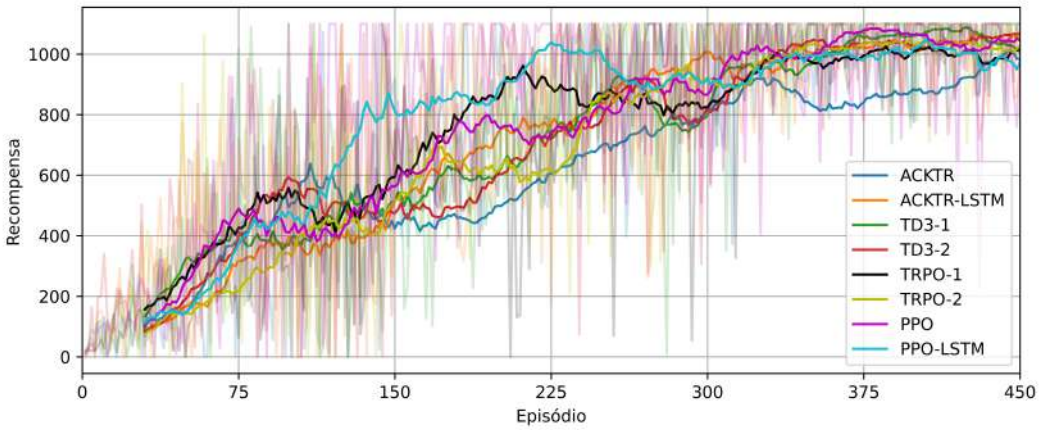
De acordo com a Figura 69, nota-se que a estrutura do controlador DBELBIC é igual à utilizada no ambiente do submarino, ou seja, o agente de DRL utiliza sinais da malha de controle para compor seus próprios sinais de estímulos *S* e *R* (arquitetura indireta).

O treinamento do DBELBIC no ambiente dinâmico do braço robótico é realizado via *Simulink*® com o agente de controle em *Python*. Os agentes de DRL escolhidos são os mesmos utilizados na abordagem do ambiente anterior, uma vez que apresentaram bom desempenho neste tipo de problema.

No que diz respeito ao mapeamento prévio da estabilidade deste ambiente, notou-se que o sinal da variável controlada (y) não é um bom candidato a compor os estímulos deste controlador, pois sua presença acaba por apresentar uma diminuição significativa na faixa da estabilidade no sistema de controle.

A Figura 70 apresenta o resultado do treinamento, obtido a partir da utilização de diferentes agentes na produção dos estímulos do DBELBIC no ambiente dinâmico do braço robótico.

Figura 70 – Treinamento do DBELBIC com diferentes agentes no ambiente do braço robótico com um grau de liberdade.



Fonte: próprio autor.

A partir de diferentes sinais do tipo degrau na referência do sistema, o agente é incentivado a minimizar o erro ao longo do tempo. O treinamento busca determinar os estímulos necessários (*S* e *R*) para alcançar o máximo de recompensas possíveis a cada episódio de 20s. Nesse caso, estipulando-se um total de 450 episódios para cada agente, o tempo médio de treinamento individual foi de *2* horas e *30* minutos.

No geral, o treinamento dos agentes de DRL do DBELBIC no ambiente do braço robótico apresentou êxito no que diz respeito ao acúmulo das recompensas (Figura 70). Neste caso, optou-se por utilizar o agente PPO, uma vez que obteve um bom desempenho no treinamento e, além disso, a partir de testes prévios, este agente obteve um funcionamento mais estável em comparação aos outros agentes. Assim sendo, utilizando-se o mapeamento prévio da estabilidade e fazendo uso do agente PPO, os sinais *S* e *R* apresentam as seguintes definições

$$S_{arm} = K_1 e + K_2 \int y + K_3, \tag{46}$$

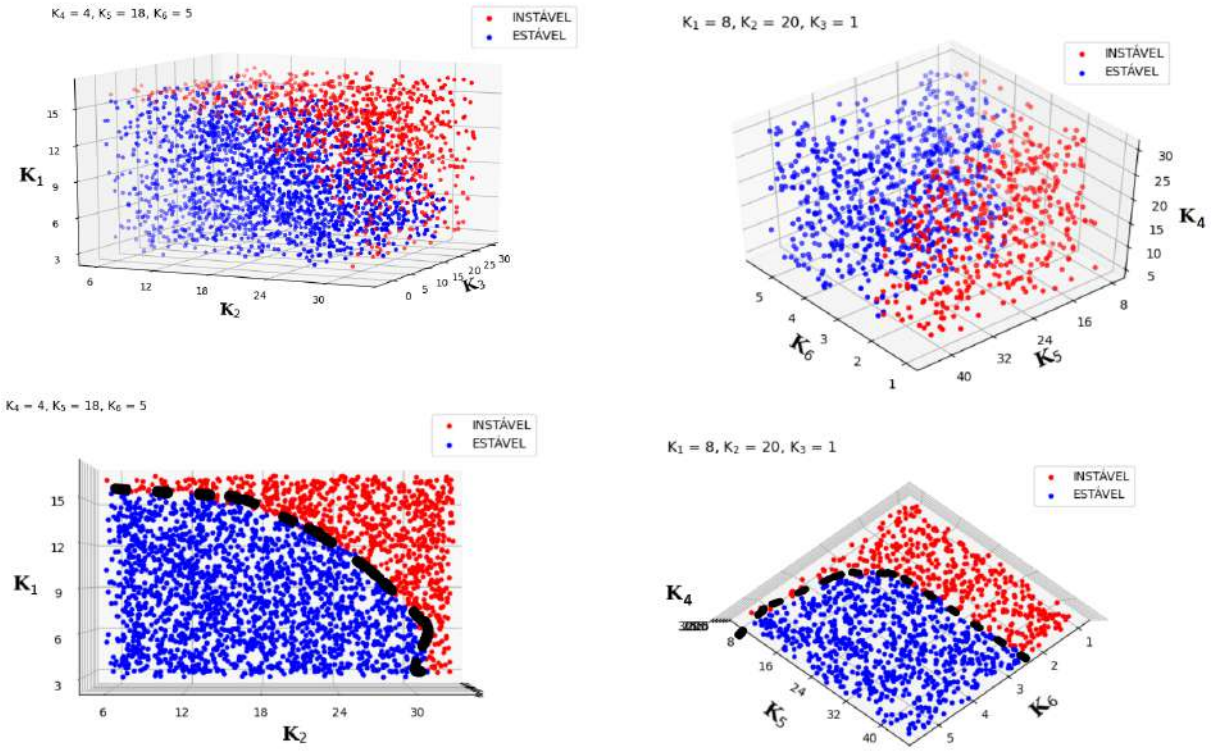
$$R_{arm} = K_4 e + K_5 \int e + K_6 \frac{de}{dt}, \tag{47}$$

onde $K_1, K_2\ K_3$, K_4, K_5 e K_6 são os ganhos provenientes das DNNs do agente de DRL, *e* é o erro, *r* a referência, *y* a variável controlada e *u* a variável de controle.

Diferentemente do que ocorreu no ambiente do submarino, o treinamento do sistema de braço robótico apresentou maiores restrições de estabilidade para a estrutura dos estímulos S e R. Desta maneira, foi necessária a realização de ajustes mais específicos nos sinais do agente de DRL de modo a tornar o sistema de controle robusto.

Uma vez obtidos os ganhos *K* nos estímulos *S* e *R* em (46) e (47), respectivamente, definidos na etapa de treinamento, apresenta-se, na Figura, 71 um mapeamento de estabilidade em alguns pontos de operação do controlador no ambiente do braço robótico.

Figura 71 – Mapeamento da estabilidade a partir dos ganhos K_1, K_2, K_3, K_4, K_5 e K_6 no ambiente do braço robótico.



Fonte: próprio autor.

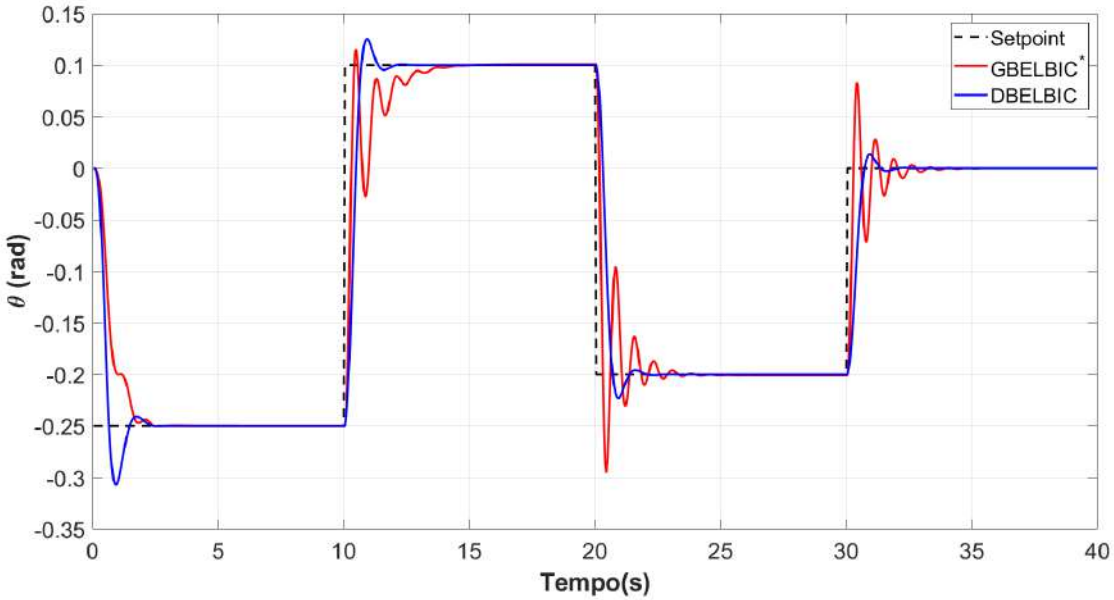
Na situação descrita na Figura 71 é visto que para cada relação de ganhos *K* existe uma restrição específica. Nesse sentido, a liberdade de ação dos agentes de DRL necessita ser mais limitada, uma vez que estímulos *S* e *R* formulados por seis ganhos está propício a apresentar uma maior tendência de instabilidade.

5.2.2.2 Resultados no ambiente do braço robótico

De maneira a realizar um comparativo adequado do DBELBIC no ambiente do braço robótico com um grau de liberdade, utilizou-se como referência o controlador emocional proposto por (LOTFI; REZAEE, 2018), o qual se utiliza de técnicas de otimização para desenvolver um tipo de controlador emocional denominado G-BELBIC.

A Figura 72 apresenta o resultado da resposta ao seguimento de referência dos controladores DBELBIC e G-BELBIC mediante sucessivas entradas tipo degrau ao sistema descrito na Figura 69.

Figura 72 – Comparativo das respostas do G-BELBIC e DBELBIC mediante a sucessivas entradas degrau junto ao sistema de braço robótico com um grau de liberdade.



Fonte: próprio autor.

A partir da Figura 72, observa-se que ambos os controladores apresentaram desempenhos satisfatórios mediante o seguimento de referência. Uma vez que o controlador G-BELBIC utilizado neste ambiente se baseia em processos de otimização, observa-se que seu desempenho pode apresentar variações quanto à sua dinâmica final. Isso se deve ao fato de que os experimentos realizados com este controlador utilizou a metodologia descrita em (LOTFI; REZAEE, 2018), todavia, alguns parâmetros foram estimados no processo de otimização.

De modo a facilitar a compreensão da Figura 72, a Tabela 10 apresenta os valores[7] da resposta dinâmica de ambos os controladores.

[7] Valores referentes ao último degrau (30s - 40s).

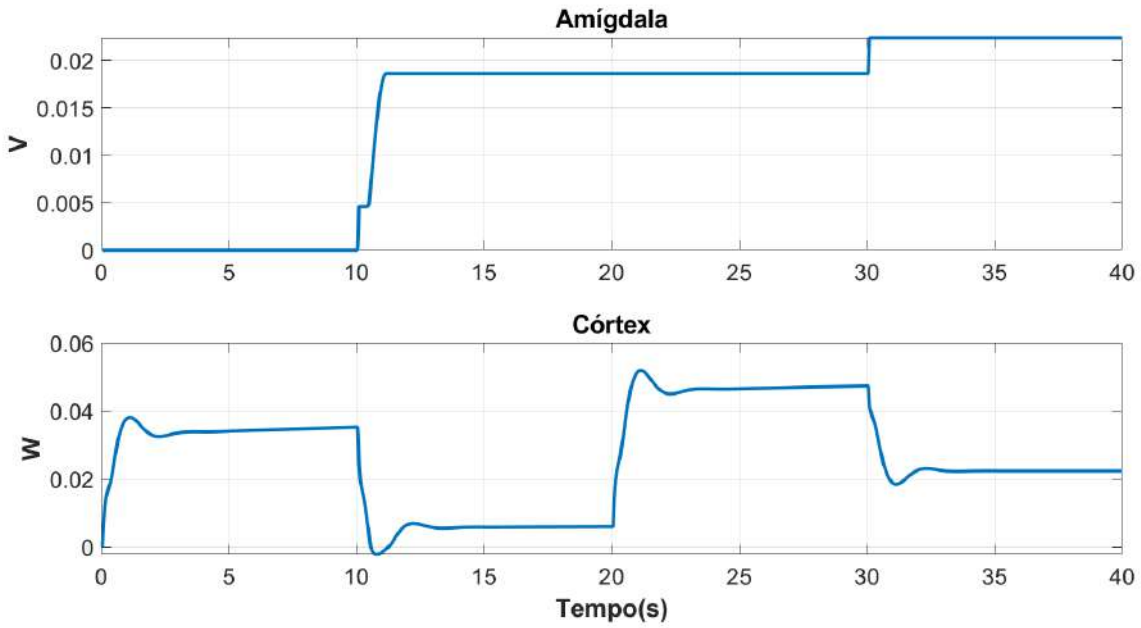
Tabela 10 – Características da respostas dinâmicas do G-BELBIC e DBELBIC aplicados ao sistema do braço robótico com um grau de liberdade.

Controlador	**Sobressinal (%)**	**Tempo de subida (s)**	**Tempo de acomodação (s)**	**Erro estacionário (%)**
G-BELBIC	48.96	0.26	4.89	0.00
DBELBIC	6.72	0.86	2.26	0.00

De acordo com a Tabela 10, nota-se que em relação ao G-BELBIC, o DBELBIC apresentou um desempenho transitório mais suave e, consequentemente, um sobressinal menor, associado a um tempo de subida maior. Este fato decorre de como foi elaborada a função de recompensa para este ambiente. Afim de simplificar o problema, as taxas de aprendizado do controlador DBELBIC (α e β) foram as mesmas utilizadas no ambiente do submarino.

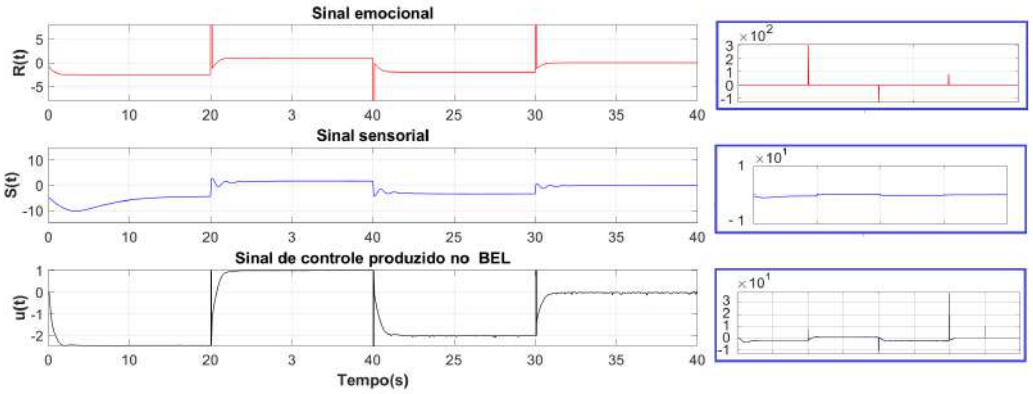
As Figuras 73 e 74 apresentam o comportamento do módulo de aprendizado do DBELBIC e os sinais produzidos neste controlador, respectivamente.

Figura 73 – Curva de aprendizado do DBELBIC junto ao sistema do braço robótico com um grau de liberdade.



Fonte:próprio autor.

Figura 74 – Sinais do DBELBIC após o treinamento do agente junto ao sistema do braço robótico com um grau de liberdade.

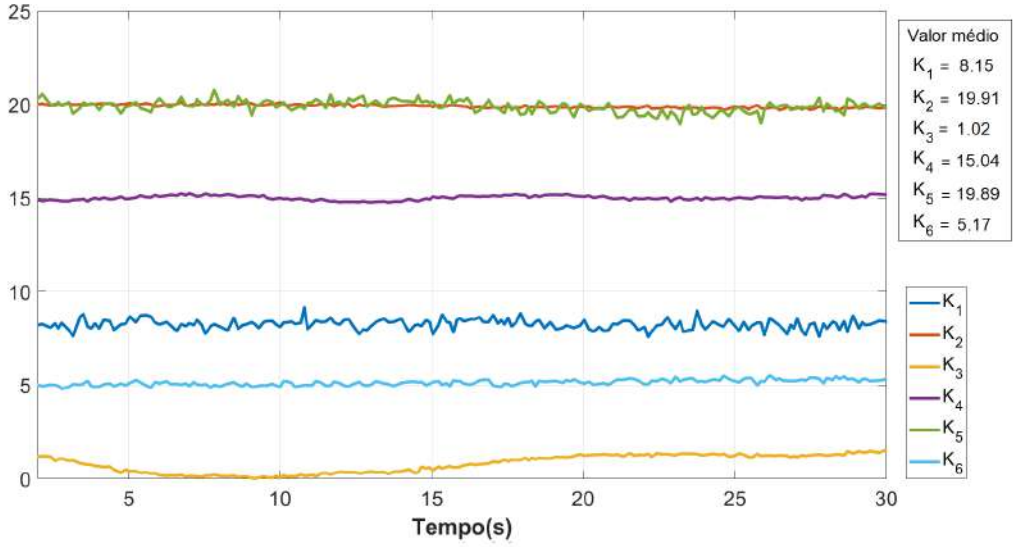


Fonte: próprio autor.

Os aprendizados da amígdala e do córtex (Figura 73), além dos sinais no DBELBIC (Figura 74), apresentaram um comportamento esperado. Nesse sentido, evidencia-se a tendência sempre crescente da amígdala e a respectiva supressão do córtex. Por outro lado, nota-se também nos sinais do DBELBIC os "picos" nas mudanças de referência do sistema.

No que diz respeito ao comportamento dos ganhos K nos estímulos S e R em (46) e (47), respectivamente, a Figura 75 apresenta esses valores ao longo do tempo de funcionamento do controlador a partir da Figura 72.

Figura 75 – Ganhos dos estímulos do DBELBIC no seguimento de referência no ambiente do braço robótico.

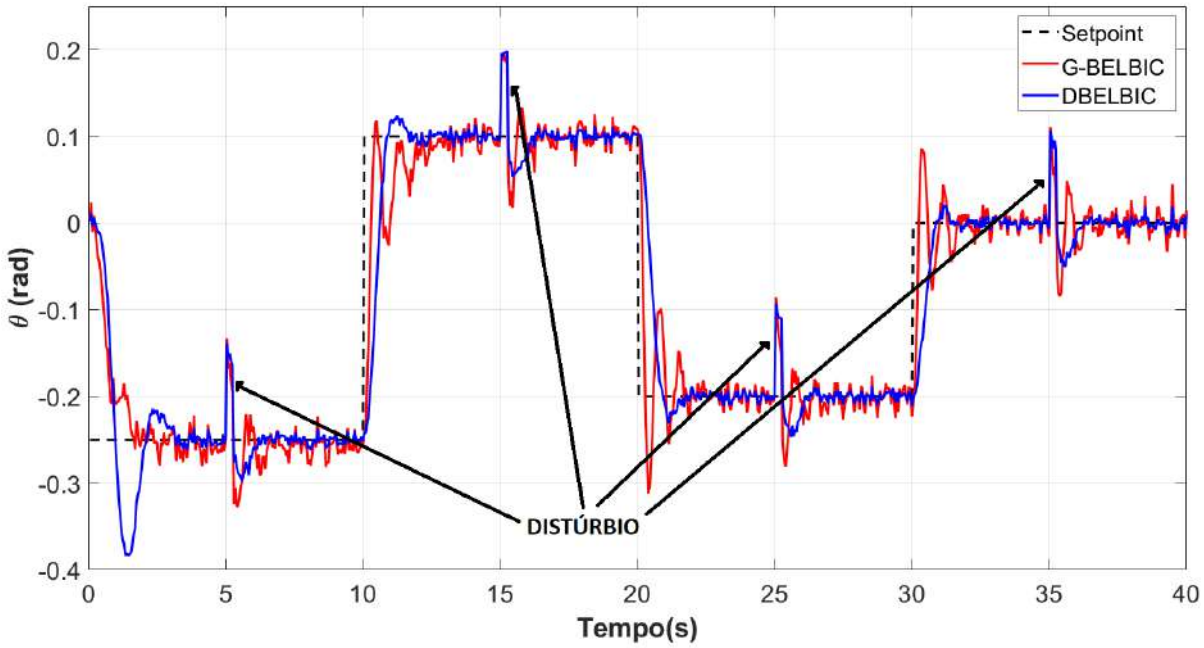


Fonte: próprio autor.

Os ganhos K (Figura 75) apresentaram uma variabilidade maior (ruídos) do que os verificados no sistema do submarino. Nesse sentido, pode ser útil a utilização de blocos saturadores nesses valores de modo a permitir um funcionamento dentro uma determinada faixa de valores de segurança.

Por fim, a análise do controlador DBELBIC no ambiente do braço robótico se concentra em sua capacidade de rejeitar ruídos e pertubações. Nesse sentido, a Figura 76 apresenta o comparativo entre os controladores G-BELBIC e DBELBIC, ambos mediante a presença de ruídos e distúrbios na saída da planta do sistema $G(s)$ (Figura 69).

Figura 76 – Controladores G-BELBIC e DBELBIC junto ao sistema do braço robótico com um grau de liberdade na presença de pertubações e ruídos.



Fonte: próprio autor.

A partir da Figura 76 é visto que os controladores apresentaram uma boa característica de rastreamento da referência, apesar da presença dos ruídos e distúrbios na planta do sistema. A Tabela 11 apresenta um comparativo dos índices da performance do erro na operação dos controladores G-BELBIC e DBELBIC, descrita na Figura 76.

Tabela 11 – Índices de desempenho do G-BELBIC e DBELBIC junto ao sistema do braço robótico com um grau de liberdade associado a ruídos e distúrbios.

Controlador	**ISE**	**IAE**	**ITAE**
G-BELBIC	0.185	1.950	63.040
DBELBIC	0.285	1.991	56.520

A partir da Tabela 11, é possível notar que o controlador DBELBIC apresentou bom desempenho nos índices analisados. Apesar do início de seu funcionamento (Figura 76) apresentar uma maior divergência no seguimento de referência, comparando-se ao mesmo período verificado no controlador G-BELBIC, seu desempenho global foi melhor.

Além da análise da robustez do controlador proposto, busca-se obter uma única função de transferência do sistema, composto pelo DBELBIC e a planta do braço robótico. Esta função deve apresentar uma dinâmica equivalente à do sistema original, permitindo aproximar e facilitar sua análise de estabilidade, assim como realizado anteriormente no sistema de submarino.

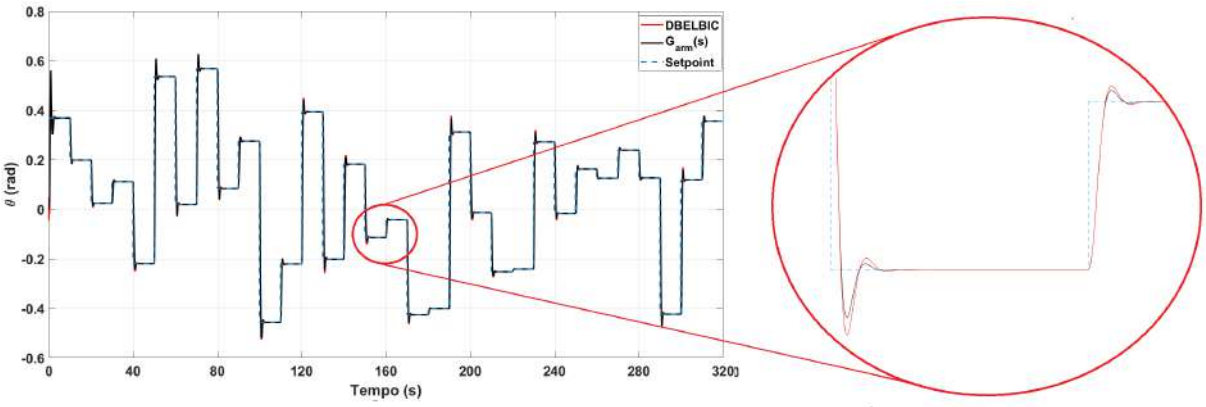
A partir das ferramentas de identificação de sistemas dinâmicos disponíveis no *software MATLAB®*, obteve-se uma função de transferência equivalente ao sistema original (Figura 69). A equação da função de transferência obtida é descrita por

$$G_{arm}(s) = \frac{-0.002833s^3 + 0.01053s^2 + 0.01432s + 0.002106}{s^4 + 0.7312s^3 + 0.2004s^2 + 0.03139s + 0.002107}. \tag{48}$$

De posse da função $G_{arm}(s)$, aplicam-se sinais do tipo degrau com diferentes valores ao longo do tempo e, busca-se comparar sua dinâmica com a do sistema original, formado pelo DBELBIC e o braço robótico (Figura 69).

A Figura 77 demonstra os resultados provenientes da função $G_{arm}(s)$ e do sistema de controle original (Figura 69), mediante diferentes valores de entradas do tipo degrau.

Figura 77 – DBELBIC e $G_{arm}(s)$ junto ao sistema do braço robótico com um grau de liberdade a partir de diferentes entradas do tipo degrau.

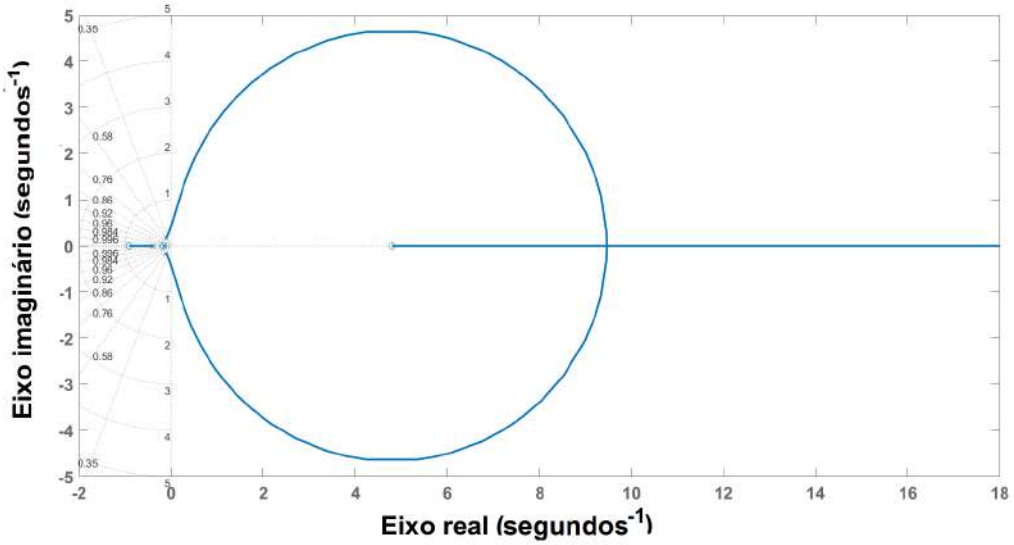


Fonte: próprio autor.

De acordo com a Figura 77, verifica-se que a função $G_{arm}(s)$ apresenta um comportamento dinâmico equivalente ao sistema formado pelo DBELBIC e o braço robótico. Uma vez obtido um sistema equivalente como em (48), é possível obter as respectivas análises usais de estabilidade para tal tipo sistema. Assim como anteriormente, utilizam-se as análises do LR e do *Diagrama de Bode* para avaliar a estabilidade do sistema descrito por $G_{arm}(s)$ em (48).

A Figura 78 apresenta o LR do sistema descrito por $G_{arm}(s)$.

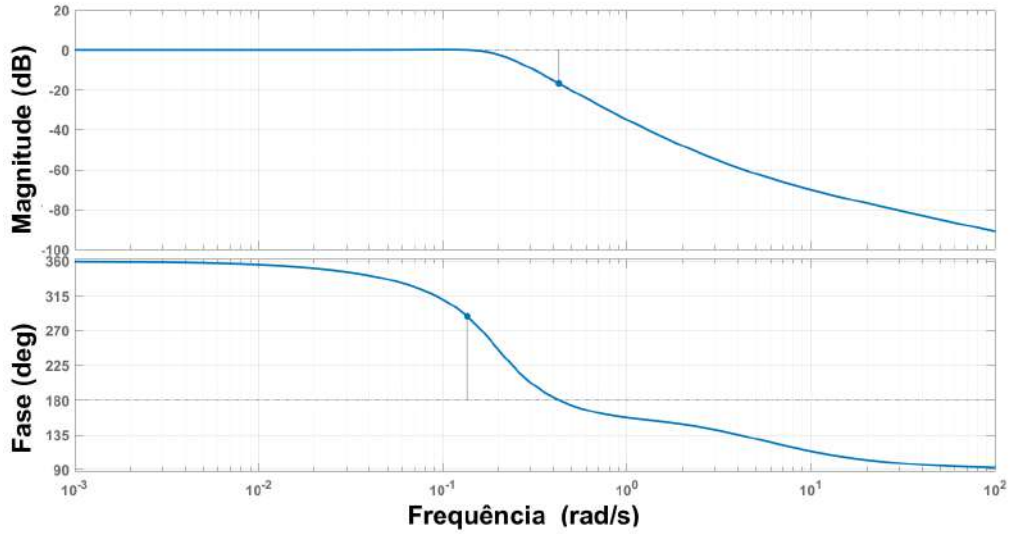
Figura 78 – LR do sistema aproximado $G_{arm}(s)$.



Fonte: próprio autor.

A partir da Figura 78, nota-se que o LR de $G_{arm}(s)$ apresenta maior predominância na região à direita do plano real, resultando assim em apenas uma pequena margem para sua estabilidade. Por fim, o *Diagrama de Bode* de $G_{arm}(s)$ é utilizado para auxiliar na interpretação da estabilidade de $G_{arm}(s)$. A Figura 79 apresenta o *Diagrama de Bode* do sistema descrito por de $G_{arm}(s)$.

Figura 79 – Diagrama de Bode do sistema aproximado $G_{arm}(s)$.



Fonte: próprio autor.

A Figura 79 demonstra que o sistema definido por $G_{arm}(s)$ apresenta uma margem de estabilidade considerável, entre as frequências 10^{-1} rad/s e 10^{0} rad/s, evidenciada nos pontos marcados nos gráficos da *Magnitude* e *Fase*.

A análise da estabilidade do sistema equivalente formado pelo DBELBIC e o braço robótico (Figura 69) evidencia que, na condição específica de arquitetura dos estímulos *S* e *R*, definidos em (46) e (47), respectivamente, o controlador não possui uma grande margem para estabilidade. De fato, a partir do mapeamento prévio da estabilidade por parte dos estímulos (Figura 71), nota-se que a margem para a variação dos ganhos *K* é mais restrita quando comparada ao caso do ambiente do submarino. Desta forma, faz-se necessário impor limites mais estreitos às variações dos valores dos ganhos *K* nas arquiteturas dos respectivos estímulos emocional e sensorial.

5.3 Problemas de regulação

Na engenharia de controle, além dos problemas de rastreamento, mencionados anteriormente, os problemas conhecidos como de *regulação* também são de grande importância prática. Neste caso, o objetivo é manter a saída do sistema constante ou em um valor predeterminado, ou seja, tornar a variação da saída próxima a zero, mesmo diante de pertubações (OGATA, 2003; DORF; BISHOP, 2018).

Nos problemas que envolvem a regulação, o valor da referência não varia com o tempo e, caso a saída do sistema não esteja no valor predeterminado inicialmente, deseja-se que retorne o mais rápido possível. De forma geral, a maior diferença entre os problemas de rastreamento e de regulação se concentra na importância do erro em regime permanente. No caso do rastreamento existe um grande interesse no erro em regime, enquanto que na regulação não há tal preocupação. Nesse sentido, pode-se considerar a regulação o caso em que se aceita um erro em regime permanente para um entrada em degrau, voltando o interesse apenas na velocidade da resposta e no sobressinal (DORF; BISHOP, 2018).

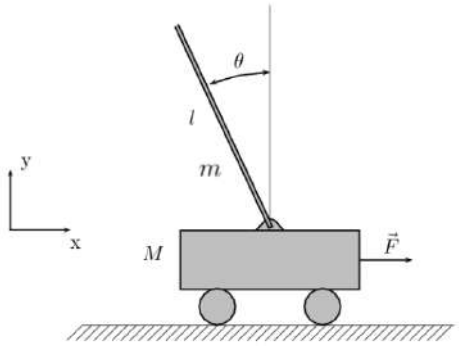
Diferentemente da abordagem utilizada neste trabalho para tratar dos problemas de rastreamento, desenvolvidos em *Simulink*®, os problemas de regulação são abordados na própria biblioteca *OpenAI Gym* do *Python*.

5.3.1 Sistema de pêndulo invertido

Uma análise importante para o controlador proposto é a sua aplicação junto ao problema do pêndulo invertido. Esse sistema é amplamente conhecido e bastante utilizado no que se refere à avaliação da performance de controladores, inclusive aqueles baseados em RL (BARTO; SUTTON; ANDERSON, 1983). A importância do pêndulo invertido está relacionada ao fato de tal sistema apresentar uma característica intrínseca de instabilidade em sua dinâmica, tornado-o útil para estudos relacionados a sistemas de controle complexos.

O sistema em questão é formado por uma barra rígida, geralmente metálica, a qual é livre para movimentar-se verticalmente em torno de um ponto fixo. A barra é conectada a um carro, o qual possui a liberdade de movimentos unidimensionais na direção horizontal, paralelo ao plano de uma pista. A Figura 80 apresenta o sistema do pêndulo invertido.

Figura 80 – Diagrama esquemático de um pêndulo invertido em um carro.



Fonte: adaptado de (GUERRERO, 2017).

O carro possui uma massa M, sujeitando-se a uma força variável F paralela à pista. No caso de haver um movimento do carro, a barra que possui uma massa m e um comprimento l, tende a cair de forma natural, considerando-se que sua posição de equilíbrio (instável) se encontra na posição vertical. O ângulo entre barra e o carro com relação a vertical é θ. Além disso, existem os atritos inerentes ao sistema, o atrito entre o carro e a pista μ_M e o atrito entre a barra e o carro μ_m e, o sistema todo submetido à força da gravidade g.

No âmbito desse sistema dinâmico, o controlador tem por objetivo aplicar uma força necessária ao carro, tal que seja possível contrabalancear a dinâmica natural do pêndulo, mantendo-o fixo na posição vertical. Diferentemente de um pêndulo normal, o qual é estável pendurado para baixo, um pêndulo invertido é inerentemente instável e, necessariamente, necessita ser constantemente forçado a permanecer na posição vertical à medida que o carro move-se na posição horizontal.

Neste trabalho, o modelo de pêndulo invertido utilizado baseia-se no trabalho de (BARTO; SUTTON; ANDERSON, 1983). As equações que descrevem as dinâmicas do ângulo da barra em relação ao carro (θ), bem como o deslocamento da posição carro (x), definem-se como

$$\ddot{\theta} = \frac{gsen\theta + cos\theta \left[\frac{-F - ml\dot{\theta}^2 sen\theta + \mu_c sgn(\dot{x})}{M+m}\right] - \frac{\mu_p \dot{\theta}}{ml}}{l\left[\frac{4}{3} - \frac{mcos^2\theta}{M+m}\right]}, \tag{49}$$

$$\ddot{x} = \frac{F + ml[\dot{\theta}^2 sen\theta - \ddot{\theta}cos\theta] - \mu_c sgn(\dot{x})}{M+m}, \tag{50}$$

onde as variáveis observadas pelo agente são a posição do carro (x), a velocidade do carro ($\dot{x}$), o ângulo da barra (θ) e a velocidade tangencial na ponta da barra ($\nu = l * \dot{\theta}$).

As Tabelas 12 e 13 apresentam as características das variáveis de observação e os parâmetros presentes em (49) e (50), respectivamente.

Tabela 12 – Características do vetor observação do ambiente do pêndulo invertido.

Ordem	Observação	Limite mínimo	Limite máximo
0	Posição do carro	-2.4 m	2.4 m
1	Velocidade do carro	-3 m/s	3 m/s
2	Ângulo da barra	-41.8°	41.8°
3	Velocidade na ponta da barra	-2 m/s	2 m/s

Tabela 13 – Informações dos parâmetros do modelo do pêndulo invertido.

Símbolo	Nome	Valor
M	Massa do carro	$1.0\ kg$
m	Massa da barra	$0.1\ kg$
l	Comprimento da barra	$0.5\ m$
μ_M	Coeficiente de atrito barra-carro	0.005
μ_m	Coeficiente de atrito carro-piso	0.002
g	Gravidade	$-9.8\ m/s^2$

No caso das recompensas, o agente recebe o valor de uma unidade (+1), a cada instante de tempo em que as variáveis da posição do carro e ângulo da barra permanecem abaixo dos limites estabelecidos para o ambiente (Tabela 12).

A equação que define o incentivo recebido pelo agente de DRL neste ambiente pode ser descrita como

$$I_{pend}(t) = (+1)(|\theta| < 12^\circ \ \& \ |x| < 2.4\,m). \tag{51}$$

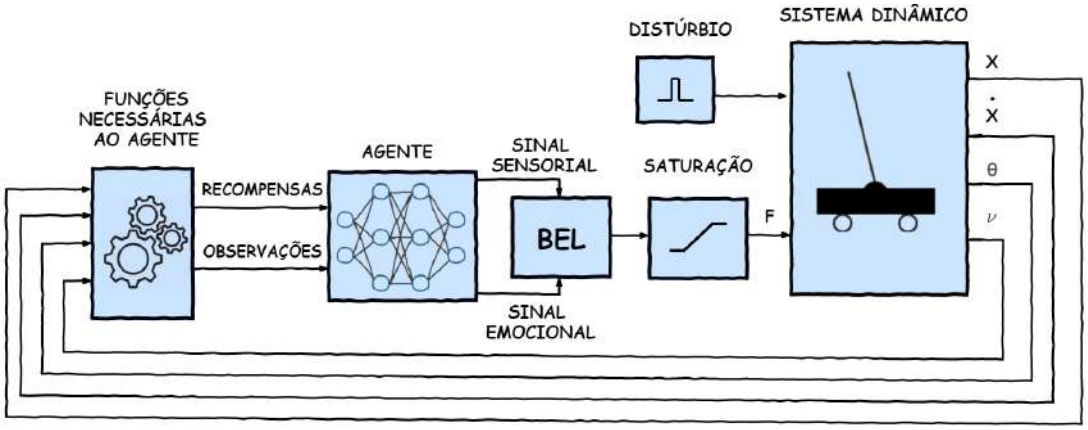
5.3.1.1 Treinamento no ambiente do pêndulo invertido

No caso dos problemas de regulação, a arquitetura de sinais nos estímulos do DBELBIC utilizada é a direta. Tal escolha é motivada pela natureza do problema e baseada em testes previamente realizados.

De modo a realizar uma análise confiável do problema, os parâmetros utilizados neste ambiente são os sugeridos pela *OpenAI Baselines* (DHARIWAL et al., 2016) (Tabela 13).

A Figura 81 apresenta o sistema de controle envolvendo o pêndulo invertido.

Figura 81 – Sistema de controle de um pêndulo invertido com DBELBIC.



Fonte: próprio autor.

A Figura 81 apresenta uma modificação na arquitetura dos sinais recebidos pelo agente, comparando-se aos problemas anteriores de rastreamento. Neste caso, o agente não se utiliza de nenhuma variável da malha de controle para compor a arquitetura dos estímulos, ou seja, o próprio agente produz o sinal como um todo (arquitetura direta).

Neste caso, busca-se avaliar o comportamento final do controlador DBELBIC diante de um problema de regulação, a despeito dos diferentes agentes de DRL utilizados nos estímulos. O sinal de controle proveniente do módulo BEL é limitado por um bloco de saturação, resultando em um valor F limitado entre $-10.0\,N$ e $10.0\,N$.

Assim como no caso dos problemas de rastreamento, busca-se utilizar o estado da arte do DRL no treinamento. Assim sendo, optou-se por fazer uso dos mesmos agentes utilizados nos problemas de rastreamento, anteriormente abordados (TD3, TRPO, PPO e ACKTR). Além disso, o projeto *RL-ZOO* (RAFFIN, 2018) é a referência para os valores dos hiperparâmetros utilizados nas arquiteturas desses agentes. Os valores dos hiperparâmetros se encontram no Apêndice B deste trabalho.

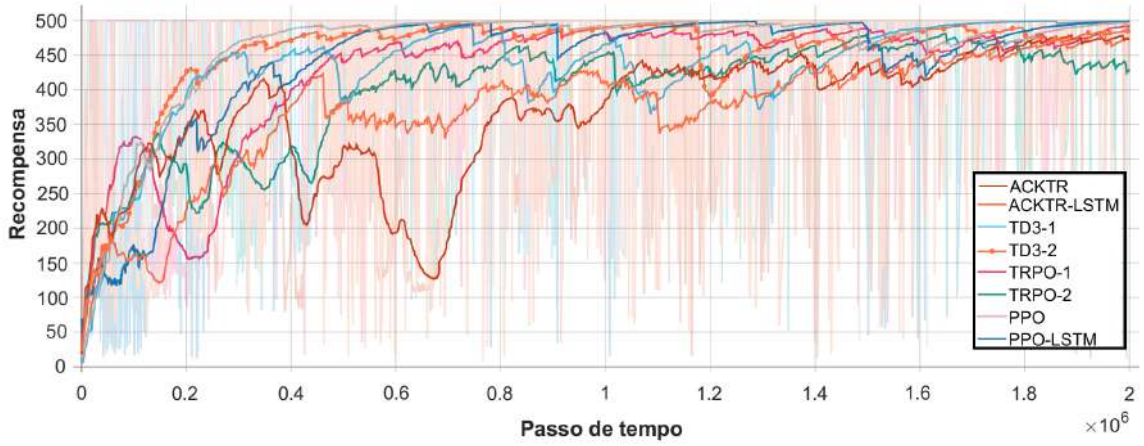
No que diz repeito à utilização de uma arquitetura direta nos estímulos *S* e *R*, verifica-se que não há um mapeamento da estabilidade referente a ganhos *K*, diferentemente do que ocorreu nos casos que envolviam a arquitetura indireta. Uma vez que o sinal da ação de controle proveniente dos agentes de DRL é diretamente aplicada ao módulo BEL não faz sentido realizar tal análise.

Uma vez que o ambiente dinâmico do sistema do pêndulo invertido não é formulado a partir do *Simulink*®, mas sim via *Python*, não foi necessário utilizar a comunicação serial entre agente e ambiente. Desta forma, a dinâmica de interação entre ambos os elementos não é realizada no "módulo de tempo real"[8]. A Figura 82 apresenta o resultado do treinamento no

[8] O *Simulink*® *Real-Time*™ executa aplicativos em tempo real no *hardware* conectado ao seu sistema físico.

ambiente do pêndulo invertido, obtido a partir da utilização de diferentes agentes na produção dos estímulos do DBELBIC.

Figura 82 – Treinamento do DBELBIC com diferentes agentes no ambiente do pêndulo invertido.



Fonte: próprio autor.

De acordo com a Figura 82, nota-se que todos os agentes utilizados no treinamento dos estímulos no ambiente do pêndulo invertido foram capazes de obter médias similares de recompensas ao final desta etapa.

Uma vez que o treinamento não é realizado em tempo real, os episódios podem apresentar diferentes durações, a depender da dinâmica agente-ambiente. Nesse caso, o critério de parada do treinamento foi estabelecido em 2 10^6 *time steps*. A Tabela 14 apresenta os valores de tempo para cada agente individualmente.

Tabela 14 – Tempo de treinamento no ambiente do pêndulo invertido para os diferentes agentes dos estímulos no DBELBIC.

AGENTE	**Tempo de treinamento**
PPO	00:17:15
PPO-LSTM	01:02:19
ACKTR	00:22:50
ACKTR-LSTM	00:47:19
TRPO-1	00:16:34
TRPO-2	00:21:22
TD3-1	00:42:23
TD3-2	00:50:14

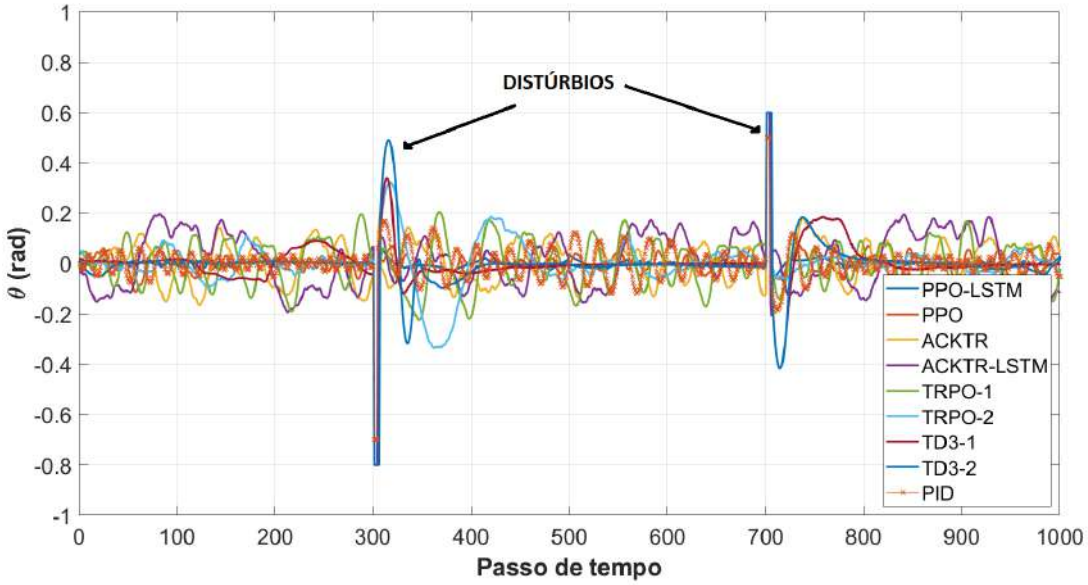
De modo distinto da análise realizada no DBELBIC nos problemas de rastreamento da referência, propõe-se avaliar todos os agentes treinados. Nesse sentido, busca-se analisar o

comportamento dos sinais de estímulos *S* e *R*, produzidos pelos diferentes agentes. Além disso, propõe-se comparar o DBELBIC formado pelos diferentes agentes com um PID[9], previamente sintonizado, analisando-se seus desempenhos neste problema de regulação.

5.3.1.2 Resultados no ambiente do pêndulo invertido

Finalizado o treinamento (Figura 82), realiza-se a análise de desempenho do DBELBIC resultante de cada agente. A Figura 83 apresenta a dinâmica deste ambiente na presença de distúrbios, utilizando-se como controladores um PID e o DBELBIC com os diferentes agentes.

Figura 83 – Comportamento da posição angular do pêndulo invertido mediante distúrbios utilizando PID e agentes de DRL nos estímulos do DBELBIC.



Fonte: próprio autor.

De acordo com a Figura 83, nota-se que o desempenho do PID e do DBELBIC associado a diferentes agentes apresentaram dinâmicas variadas na regulação do sistema. Todavia, todos foram capazes de suprimir os distúrbios presentes na posição angular do pêndulo. No entanto, é importante destacar que no caso da utilização do PID, o distúrbio aplicado foi de menor intensidade ($\Delta\theta_1 = -0.7; \Delta\theta_2 = 0.5$), quando comparado ao caso do DBELBIC ($\Delta\theta_1 = -0.8; \Delta\theta_2 = 0.6$). Isso se deve ao fato que nessa condição de sintonia, o PID não foi capaz de estabilizar o pêndulo na faixa requerida.

A Tabela 15 apresenta os valores dos índices de erros[10] para cada caso individualmente.

[9] P = 0.3, I = 0.006 e D= 0.5.

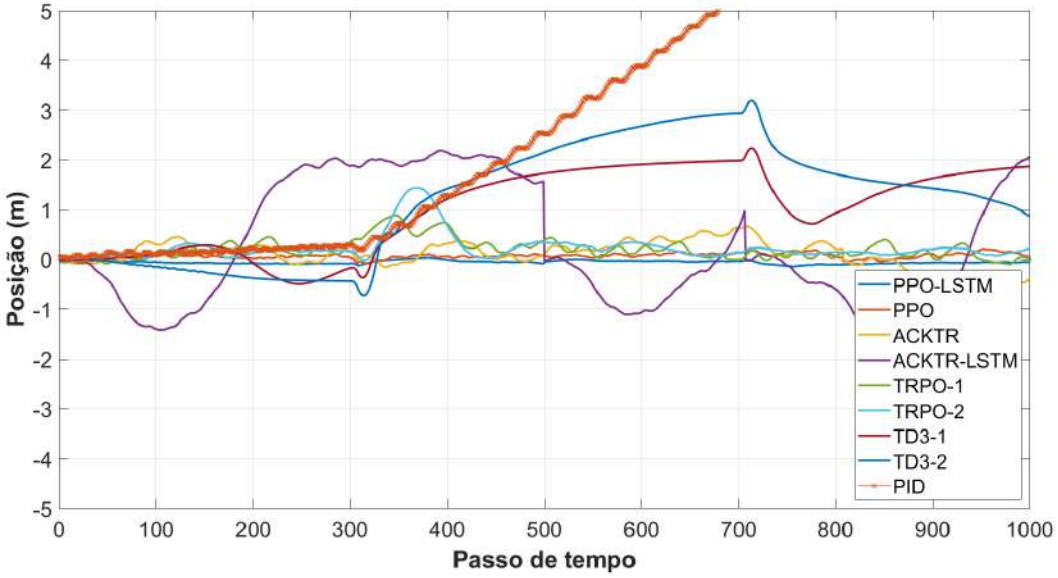
[10] Valores referentes ao erro da posição angular em torno do 0^0.

Tabela 15 – Índices de desempenho do PID e DBELBIC associado aos agentes de DRL no sistema do pêndulo invertido mediante a distúrbios na posição angular.

Controladores	ISE	IAE	ITAE
PID	12.90	45.42	$2.55\ 10^4$
DBELBIC			
PPO	4.97	29.28	$2.09\ 10^4$
PPO-LSTM	4.12	13.23	$1.55\ 10^4$
ACKTR	10.52	75.29	$2.28\ 10^4$
ACKTR-LSTM	15.78	101.65	$5.79\ 10^4$
TRPO-1	14.93	88.36	$9.57\ 10^3$
TRPO-2	13.21	58.66	$1.96\ 10^4$
TD3-1	11.19	45.47	528.04
TD3-2	12.71	41.17	$1.22\ 10^4$

A partir de observações da Figura 83 e da Tabela 15, nota-se que o DBELBIC associado aos agentes PPO e PPO-LSTM apresentaram os melhores desempenhos na regulação do sistema. Por outro lado, além da análise da posição angular, busca-se observar a variável da posição do carro mediante os distúrbios no ângulo do pêndulo. A Figura 84 apresenta o comportamento da posição do carro sob os distúrbios no ângulo de pêndulo.

Figura 84 – Comportamento da posição do carro no sistema do pêndulo invertido mediante distúrbios utilizando PID e agentes de DRL nos estímulos do DBELBIC.



Fonte: próprio autor.

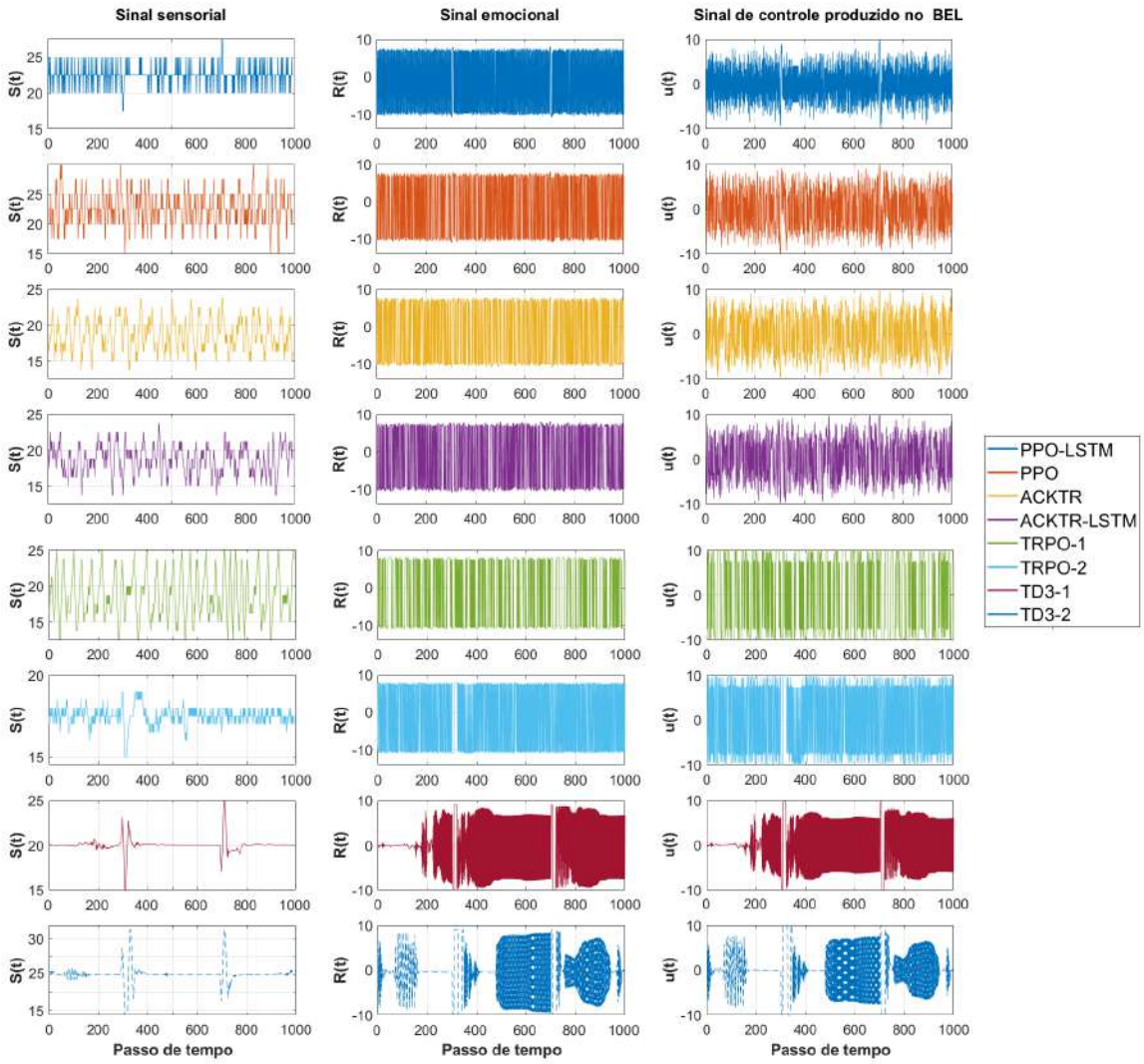
A posição do carro (x) mediante distúrbios na posição angular (θ) do pêndulo (Figura 84) apresentou diferentes comportamentos a depender do controlador. Neste caso, nota-se que existe uma maior variação da posição do carro em relação a posição inicial nos casos do DBELBIC com

os agentes ACKTR-LSTM, TD3-1, TD3-2 e, principalmente o PID, onde a partir do primeiro distúrbio a posição do carro foi sempre crescente.

No que se refere aos sinais de estímulos *S* e *R* fornecidos ao DBELBIC a partir dos agentes de DRL, nota-se que cada agente apresenta uma característica dinâmica particular. Nesse caso, o comportamento de aprendizado do DBELBIC varia de acordo com o agente. Além disso, de forma a simplificar a implementação do controlador neste ambiente, definiram-se as contantes de aprendizados do DBELBIC (α e β) com valores constantes iguais a 0.025.

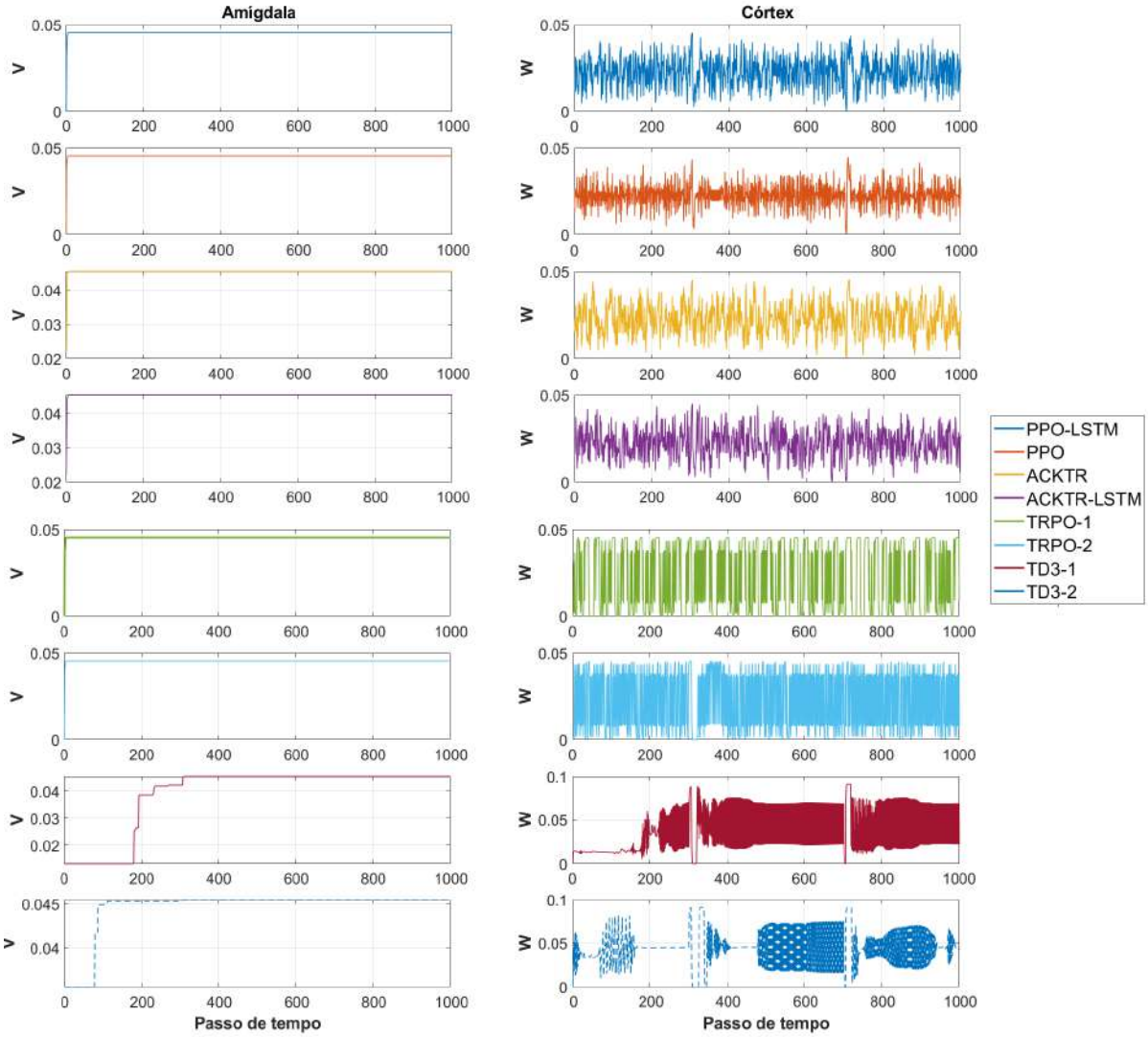
As Figuras 85 e 86 apresentam, respectivamente, os sinais e aprendizados do DBELBIC para cada tipo de agente de DRL, referindo-se ao comportamento das Figuras 83 e 84.

Figura 85 – Sinais do DBELBIC em relação aos agentes de DRL no ambiente do pêndulo invertido mediante distúrbios na posição angular.



Fonte: próprio autor.

Figura 86 – Curvas de aprendizado do DBELBIC para os diferentes tipos de agentes de DRL mediante distúrbios na posição angular do sistema de pêndulo invertido.



Fonte: próprio autor.

O comportamento dos sinais sensorial e emocional, produzidos no ambiente do pêndulo invertido (Figura 85) apresentam características semelhantes, podendo-se observar uma pequena variação na dinâmica de tais sinais nos momentos dos distúrbios. Os agentes que apresentaram maior diferença de comportamento entre os demais são o TD3-1 e TD3-2, os quais possuem uma menor intensidade na geração de tais estímulos.

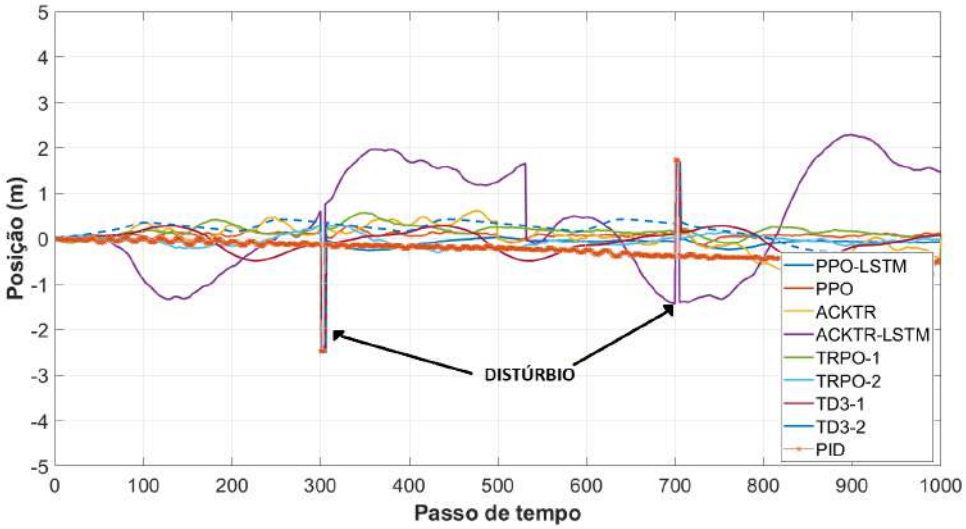
No caso dos aprendizados do DBELBIC (Figura 86), nota-se que o córtex possui uma característica semelhante ao sinal emocional. Uma vez que a amígdala tem um comportamento praticamente constante positivo, o córtex atua para mitigar essa "emoção" e proporcionar um sinal de controle adaptável.

Outra análise proposta neste contexto é, diferentemente do caso anterior (distúrbio na

posição angular), a aplicação de distúrbios na variável da posição do carro, buscando-se observar o comportamento da posição angular do sistema.

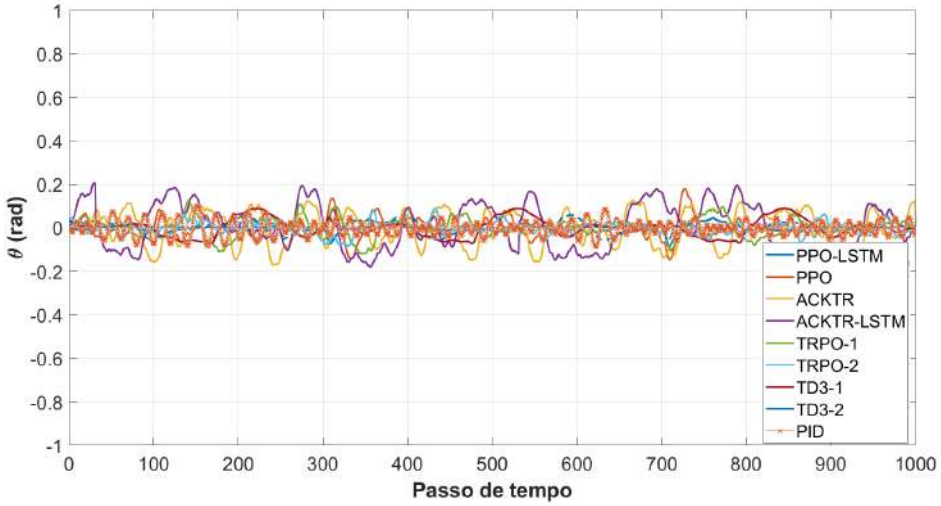
As Figuras 87 e 88 apresentam o comportamento da posição do carro e da posição angular do pêndulo, respectivamente, mediante a distúrbios presentes na posição do carro.

Figura 87 – Comportamento da posição do carro no sistema do pêndulo invertido mediante distúrbios na posição do carro utilizando PID e agentes de DRL nos estímulos do DBELBIC.



Fonte: próprio autor.

Figura 88 – Comportamento da posição angular do pêndulo invertido mediante distúrbios na posição do carro utilizando PID e agentes de DRL nos estímulos do DBELBIC.



Fonte: próprio autor.

A partir da observação das Figuras 87 e 88, é possível perceber que diante de distúrbios ($\Delta x_1 = -2.5; \Delta x_2 = 1.5$) na posição do veículo, o efeito geral na posição angular do pêndulo é pouco percebido. A Tabela 15 apresenta os valores dos índices de erros[11] para cada caso individualmente, referindo-se à situação descrita na Figura 88.

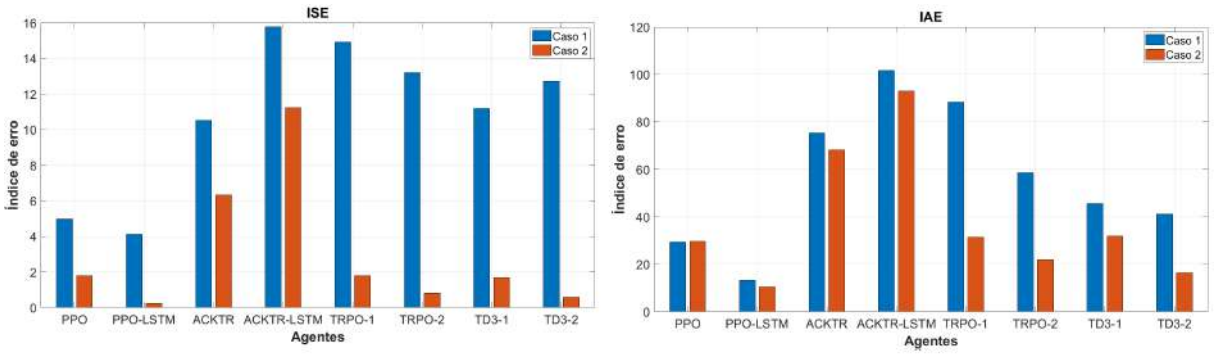
Tabela 16 – Índices de desempenho do PID e DBELBIC no sistema do pêndulo invertido mediante a distúrbios na posição do carro.

Controladores	ISE	IAE	ITAE
PID	1.87	31.03	1.71 10^4
DBELBIC			
PPO	1.79	29.67	2.23 10^4
PPO-LSTM	0.25	10.43	1.66 10^4
ACKTR	6.33	68.19	1.49 10^4
ACKTR-LSTM	11.23	93.10	5.59 10^3
TRPO-1	1.79	31.45	8.02 10^3
TRPO-2	0.80	22.01	7.95 10^3
TD3-1	1.69	31.78	679.41
TD3-2	0.57	16.42	3.60 10^4

De acordo com os dados da Tabela 16, nota-se que os agentes que obtiveram um melhor desempenho em relação aos índices de erro foram o PPO-LSTM e TD3-2. Nesse caso, o objetivo dos controladores foi apenas de manter estável a variável da posição angular do pêndulo, mesmo que a posição do carro sofresse com distúrbios externos.

De forma a comparar os desempenhos do DBELBIC mediante os distúrbios na posição angular e na posição do carro, utiliza-se como notação "caso 1" e "caso 2" , receptivamente. De modo semelhante ao caso 1, os distúrbios aplicados no caso 2 junto ao PID foram de menor intensidade. A Figura 89 apresenta os valores dos índices IAE e ISE para os casos 1 e 2 avaliados.

Figura 89 – Índices de erros dos agentes de DRL no DBELBIC para os casos 1 e 2.



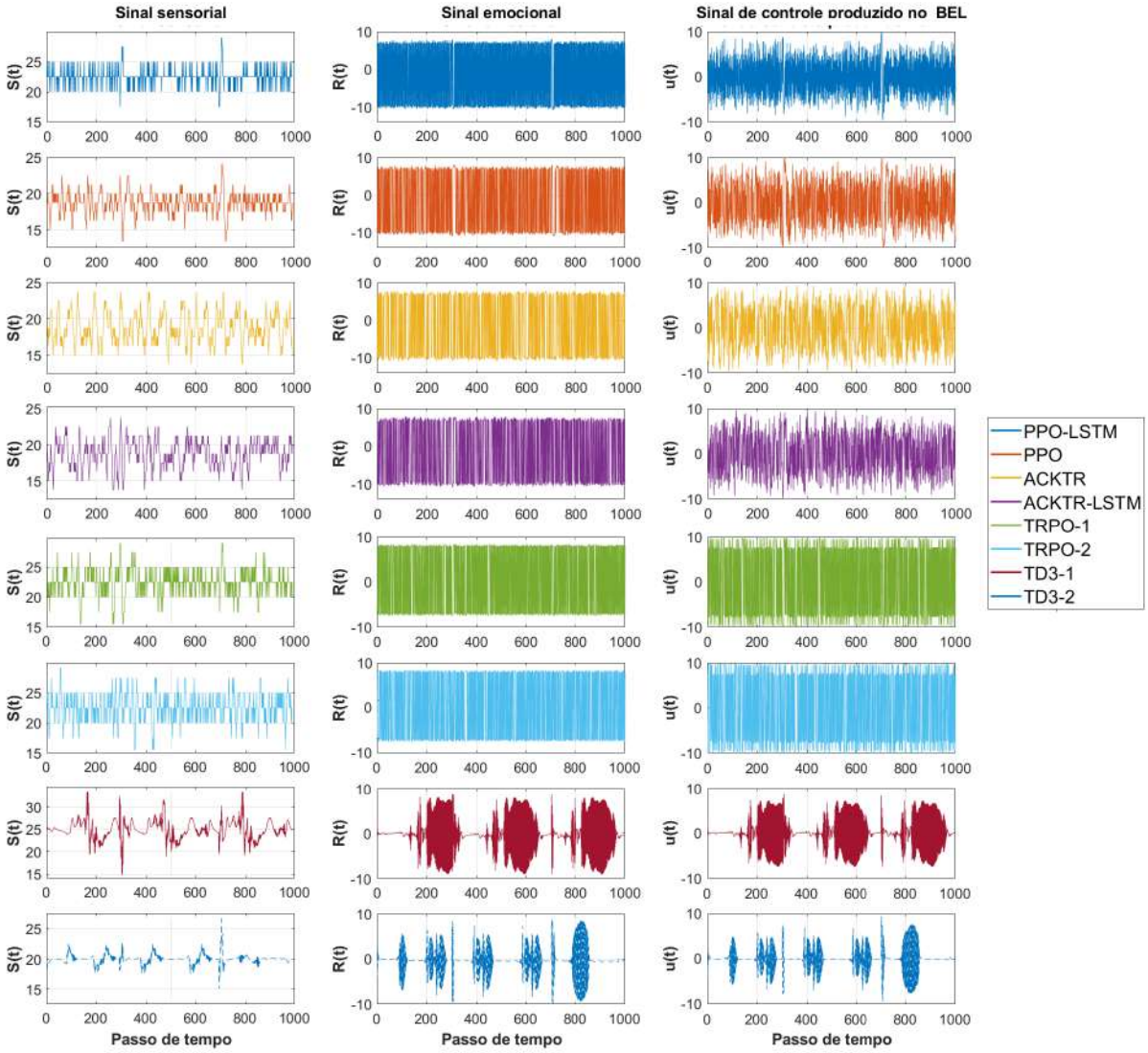
Fonte: prórpio autor.

[11] Valores referentes ao erro da posição angular em torno do 0^o.

De fato, os índices de erro associados ao caso 2 (Figura 89) apresentam menores valores, pois os distúrbios na posição do carro não geram maiores efeitos no comportamento angular do pêndulo. Além disso, é possível notar que em ambas as situações, o agente PPO-LSTM apresentou os menores valores entre os índices de erro avaliados.

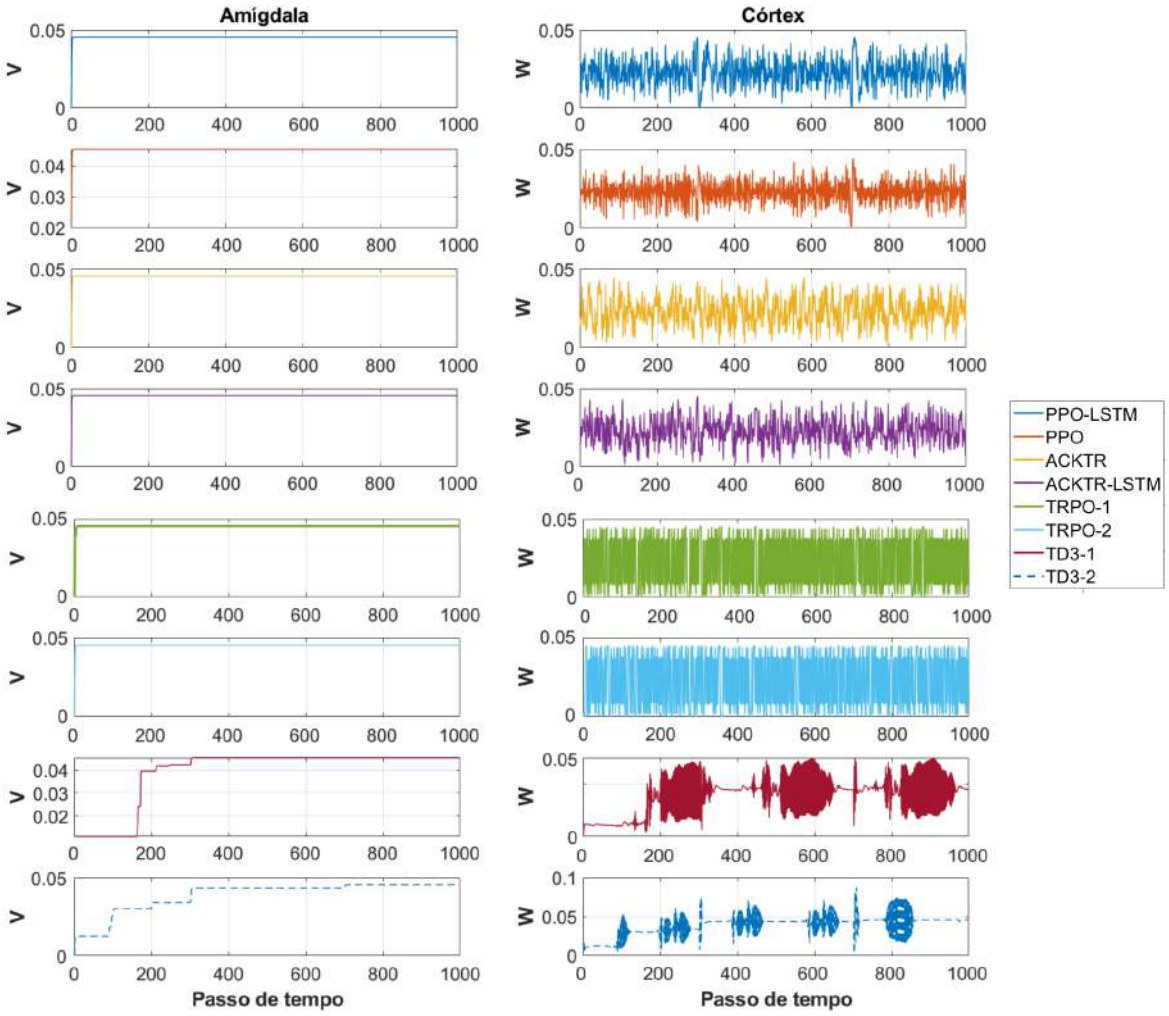
No que se refere ao caso 2, as Figuras 90 e 91 apresentam os sinais e aprendizados do DBELBIC para cada tipo de agente de DRL, respectivamente, referindo-se ao comportamento das Figuras 87 e 88.

Figura 90 – Sinais do DBELBIC em relação aos agentes de DRL no ambiente do pêndulo invertido diante de distúrbios na posição do carro.



Fonte: próprio autor.

Figura 91 – Curvas de aprendizado do DBELBIC para os diferentes tipos de agentes de DRL mediante distúrbios na posição do carro.



Fonte: próprio autor.

A partir da Figura 90, é possível notar semelhanças no comportamentos dos sinais emocionais, sensoriais e de controle, quando comparados aos da situação observada no caso 1 (Figura 85). Por outro lado, a respeito dos aprendizados da amígdala e do córtex no DBELBIBC nesta situação, também existem semelhanças (Figura 91).

Apesar das semelhanças observadas nos casos 1 (Figuras 85 e 86) e 2 (Figuras 90 e 91), percebe-se que houve uma variação maior dos sinais no caso dos agentes TD3-1 e TD3-2, tanto a respeito dos sinais quanto nos valores dos aprendizados.

De forma geral, o problema de rastreamento utilizando o ambiente do pêndulo invertido proporcionou uma situação adequada para a análise do comportamento dos estímulos no DBELBIC através de uma arquitetura direta. Além disso, foi possível perceber uma distinção comportamental dos aprendizados da amígdala e do córtex. O aprendizado da amígdala não

apresentou uma tendência de variação constante em seu crescimento, assim como observado nos casos dos problemas de rastreamento. Por outro lado, o aprendizado do córtex demonstrou uma característica similar aos sinais produzidos no estímulo emocional.

De fato, os sinais produzidos no DBELBIC a partir de uma arquitetura direta proporcionaram uma sensibilidade maior a este controlador, evidenciado no comportamento oscilatório, mas adequado para a regulação do sistema do pêndulo invertido. Apesar da sensibilidade do controlador proposto, nota-se que este apresentou uma melhora na regulação do sistema quando comparado a um controlador PID tradicional.

5.4 DBELBIC em sistemas industriais

A última análise proposta para o DBELBIC neste trabalho consiste na sua utilização prática em sistemas industriais. Dessa forma, busca-se realizar uma avaliação quanto ao seu aspecto funcional em um ambiente dinâmico real.

De forma geral, simulações constituem a maneira mais comum para o treinamento dos agentes de DRL. Uma vez que se tenha um modelo adequado do ambiente dinâmico, pode-se realizar simulações mais rapidamente do que o funcionamento no tempo real e, além disso, tem-se um controle maior a respeito das condições dinâmicas do processo. Nesse sentido, a motivação do treinamento em simulação se dá pelo fato do processo de aprendizado exigir muitas interações do agente com o ambiente, ou seja, tentativas, erros e correções. Por esta razão, pode-se levar milhares ou milhões de episódios para convergir para uma solução ideal em se tratando do DRL (SUTTON, 1984).

Por outro lado, faz-se evidente a necessidade de abordar tais controladores com base em DRL de forma prática, pois geralmente pretende-se realizar o controle de processos físicos reais. Desta forma, diferentemente dos casos anteriormente abordados neste trabalho, utiliza-se em primeiro lugar um ambiente simulado para a realização do treinamento do agente de DRL e por fim, faz-se o *deployment*[12] do controlador proposto em um protótipo embarcado.

Na prática, apesar do treinamento do agente ser realizado em simulação (*offline)*, tem-se a necessidade de continuar seu treinamento no *hardware* físico após o processo do *deployment* (*online*). Tal fato se dá pela necessidade do agente em se adaptar às imprecisões do modelo utilizado em simulação. Além disso, geralmente os ambientes reais apresentam, ao decorrer do tempo, pequenas variações em sua dinâmica, exigindo assim que o agente continue a aprender ocasionalmente, ajustando-se a tais mudanças.

5.4.1 Sistema de exaustão industrial

De maneira a avaliar o funcionamento prático, bem como o desempenho do DBELBIC em ambiente real, utiliza-se como parte desta análise uma planta protótipo de um sistema de

[12] Do inglês implementação.

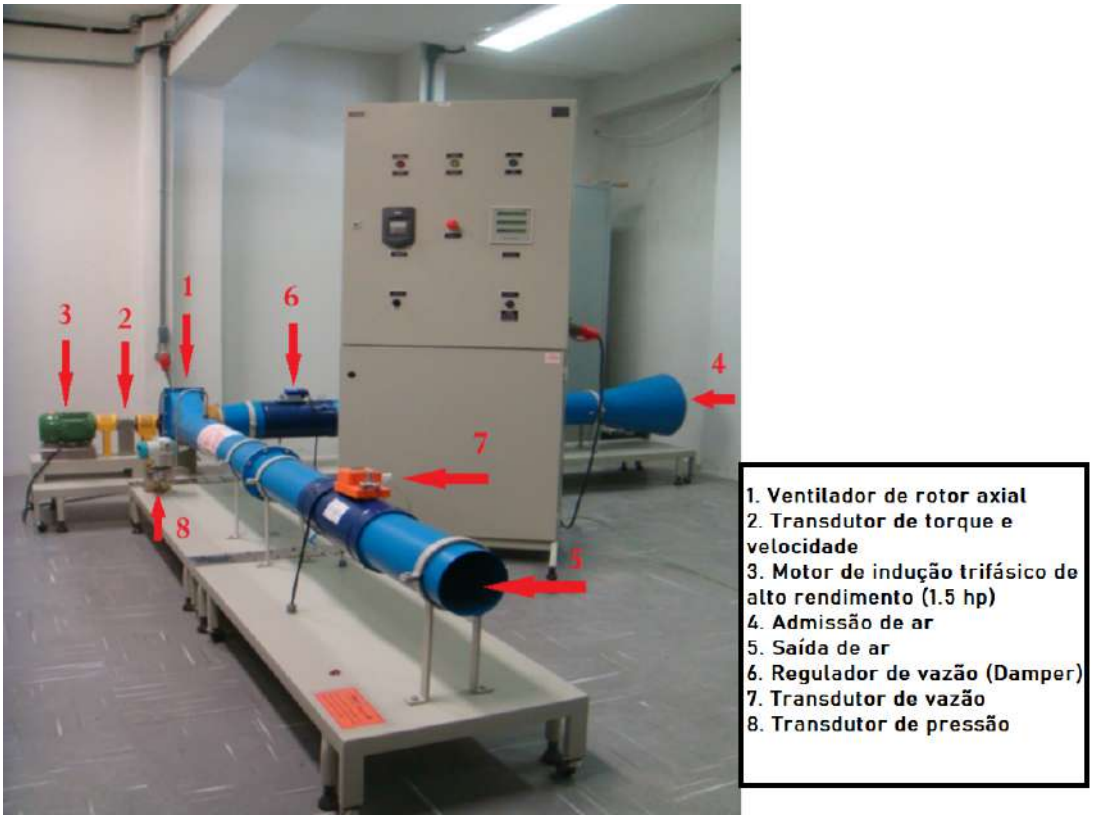
exaustão industrial do LAMOTRIZ.

A motivação para escolha desse sistema de exaustão industrial é dada pela possibilidade de comparação do controlador DBELBIC com outros controladores utilizados neste ambiente. Em resumo, utiliza-se neste ambiente dinâmico um controlador PID tradicional e outro baseado em inferência (*Fuzzy*) ou controlador baseado em lógica difusa - *fuzzy logic controller* (FLC). Neste caso, destaca-se a utilização do controlador *Fuzzy* aplicado a este sistema de exaustão, o qual permitiu obter resultados concernentes ao desenvolvimento de aplicações com inteligência artificial em sistemas motrizes associados (SILVA et al., 2019).

Os controladores associados à planta do exaustor industrial do LAMOTRIZ proporcionam um ambiente ideal para a realização de um comparativo prático da proposta do DBELBIC, uma vez que são sistemas em execução bem avaliados e consolidados.

A Figura 92 apresenta o protótipo do sistema de exaustão industrial localizado no LAMOTRIZ.

Figura 92 – Sistema de exaustão industrial no LAMOTRIZ.



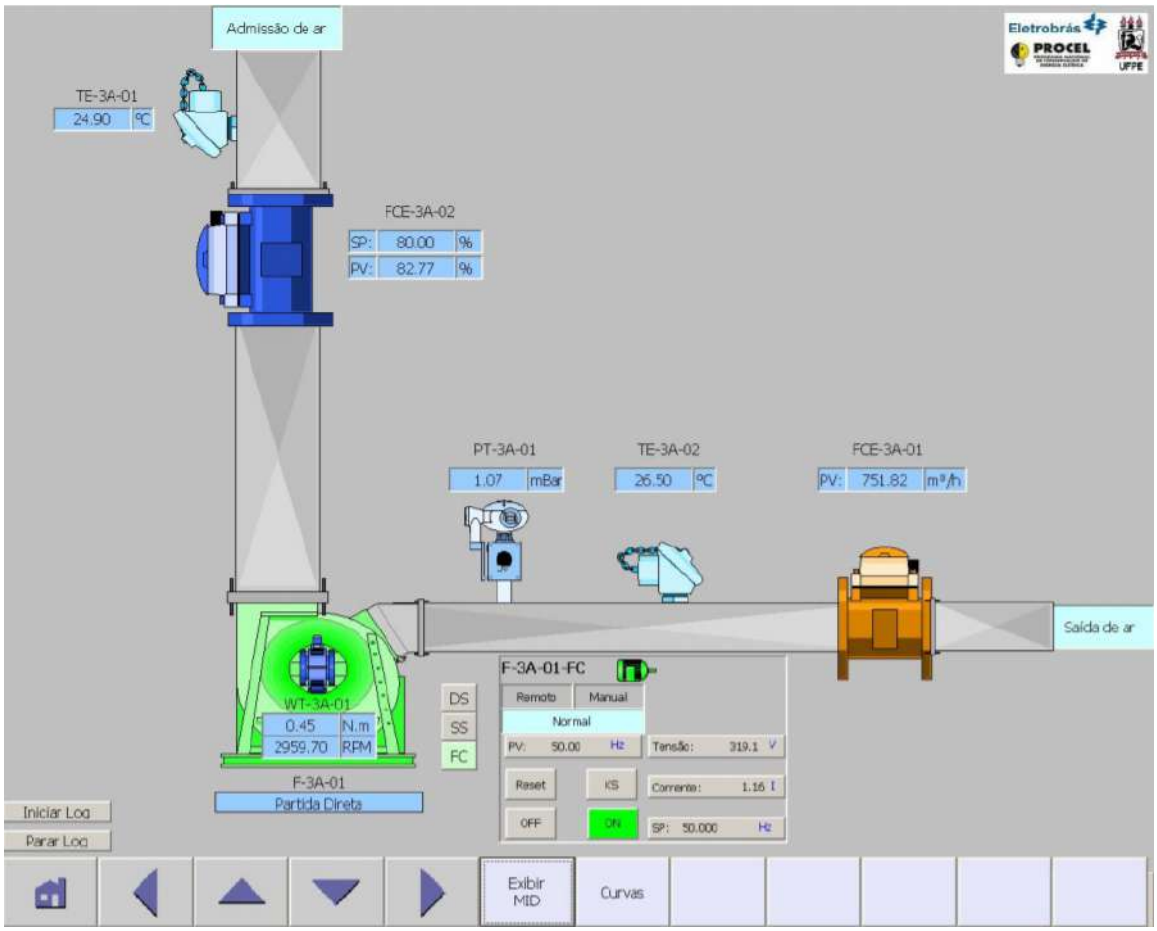
Fonte: próprio autor.

O protótipo descrito na Figura 92 é constituído por um conjunto de automação, o qual é utilizado para a realização de estudos de eficiência energética associados ao processo de exaustão industrial.

A rede desse sistema faz uso de uma comunicação do tipo PROFIBUS-DP para realizar a conexão entre o CLP e os sistemas de medição e proteção associados. Além disso, utiliza-se uma interface INDUSTRIAL-ETHERNET para comunicar o CLP e o supervisório do sistema.

A Figura 93 apresenta o supervisório do sistema de exaustão industrial do LAMOTRIZ desenvolvido em *WinCC*®.

Figura 93 – Supervisório do sistema de exaustão industrial em *WinCC*®.

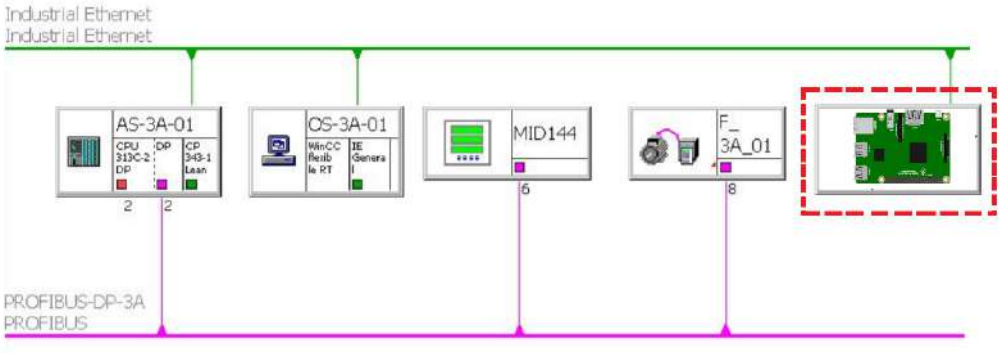


Fonte: próprio autor.

Neste trabalho, o controlador proposto é associado a um *hardware Raspberry Pi*, conectando-se a rede do sistema por meio da linha INDUSTRIAL-ETHERNET.

A Figura 94 apresenta o esquema da rede associada ao sistema de exaustão industrial utilizado neste trabalho.

Figura 94 – Rede de comunicação do sistema de exaustão industrial no LAMOTRIZ.



Fonte: adaptado de (EQUITRON SISTEMAS, 2012).

De acordo com a Figura 94, nota-se que a inserção do *Raspberry Pi* (à direita) apresenta a possibilidade de comando e supervisão do CLP/supervisório via rede *Wireless*, justificada pela existência da infraestrutura de rede no LAMOTRIZ. Desta maneira, possibilita-se a adequação do sistema de exaustão a *IoT*[13], uma vez que o *Raspberry Pi* dispõe de vários recursos de comunicação associados (RASPBERRY, 2019; BAUERMEISTER, 2019).

A Figura 95 apresenta a ligação física da *Raspberry Pi* com os periféricos e a linha INDUSTRIAL-ETHERNET do laboratório.

Figura 95 – Conexões *Raspberry Pi*.

Fonte: próprio autor.

O conjunto do sistema utiliza em sua automação um CLP SIMATIC® S7-300 Siemens®. Por este motivo, utiliza-se o software STEP7® para o desenvolvimento e gerenciamento do

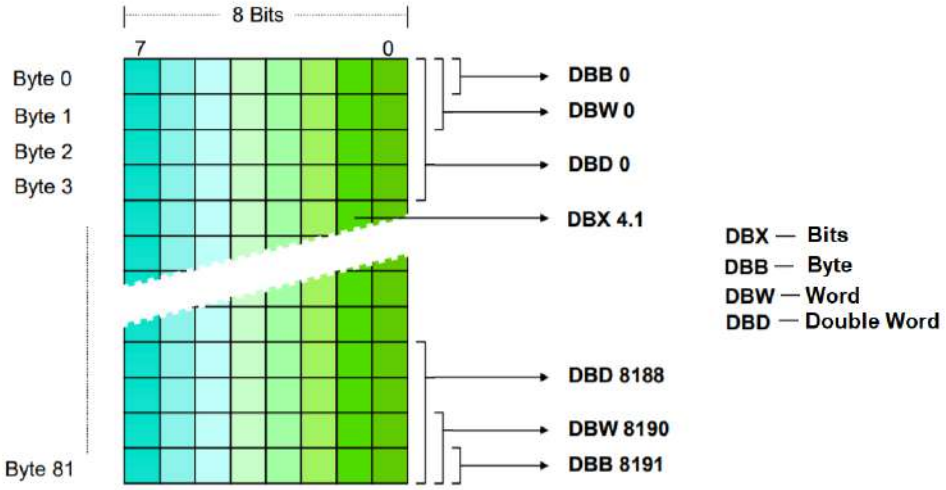
[13] Do inglês Internet of Things ou Internet das Coisas.

sistema. Nesse sentido, faz-se necessário o conhecimento da estrutura de armazenamentos dos dados no CLP.

Neste trabalho, uma vez que se utiliza a biblioteca *Python* Snap7 para a comunicação entre o *hardware Raspberry Pi* e o CLP, deve-se identificar corretamento o endereçamento das informações na estrutura das DBs[14]. As DBs são utilizadas para armazenar dados utilizados no programa do CLP, os quais ocupam espaço na memória do *hardware*.

A Figura 96 ilustra a representação da estrutura das DBs utilizadas no CLP Siemens®.

Figura 96 – Representação da estrutura das DBs no CLP Siemens®.



Fonte: adaptado de (UNICONTROL, 2003).

De maneira semelhante aos *bits* de memórias, as informações nas DBs são endereçadas *byte* a *byte*. No caso da CPU S7-300, utilizada neste trabalho, o comprimento máximo de uma DB é de *8 KByte*.

A fim de permitir a aplicação do controlador DBELBIC no sistema de exaustão industrial do LAMOTRIZ, deve-se atentar para as DBs que forneçam os endereços das variáveis pertinentes da malha de controle. A partir da localização correta do endereço de uma DB específica, pode-se ler ou escrever esta variável via Snap7. O código fonte da leitura e escrita envolvendo a comunicação *Raspberry Pi* e CLP via Snap7 se encontra no Apêndice A deste trabalho.

A Tabela 17 apresenta algumas DBs importantes do sistema de exaustão industrial do LAMOTRIZ no contexto deste trabalho.

[14] Do acrônimo em inglês Data Blocks ou Bloco de Dados. As DBs são memórias não voláteis para o armazenamento dos valores de variáveis no CLP.

Tabela 17 – Endereçamento das DBs do sistema de exaustão industrial no LAMOTRIZ.

Endereço	**Tipo**	**Descrição**
DB1 DBD 238	Real	Vazão real no exaustor
DB2 DBD 202	Real	Setpoint de frequência quando em modo manual
DB2 DBW 224	Inteiro	Frequência instantânea no motor
DB2 DBW 226	Inteiro	Tensão no motor
DB2 DBW 228	Inteiro	Corrente no motor
DB12 DBD 04	Real	Setpoint de referência
DB12 DBD 08	Real	Erro

No que se refere à viabilização do treinamento do sistema descrito na Figura 92, é necessário obter seu modelo dinâmico em função de transferência para a realização do treinamento *offline* em ambiente *Simulink*®. De acordo com trabalhos anteriores (SILVA, 2017), o modelo dinâmico da função de transferência em modo discreto desse sistema de exaustão é descrito por

$$H_{fan}(z) = \frac{3.773}{z - 0.338}. \tag{52}$$

A equação (52) é obtida a partir de uma discretização com taxa de amostragem de 1 *Hz* de uma função de transferência em modo contínuo, previamente concebida através do levantamento das características de entrada e saída do sistema.

O modelo em (52) admite como sinal de entrada um valor de frequência (*Hz*) no motor de indução trifásico e, a partir disso, obtém-se na saída do sistema uma vazão no duto de ventilação (m^3/h).

Assim como nos problemas de rastreamento, previamente abordados neste trabalho, o agente de DRL no DBELBIC se utiliza da *variável do erro*, *integral do erro* e *derivada do erro* para a observação deste ambiente, uma vez que a natureza do problema é a mesma.

A Tabela 18 apresenta as características do *vetor observação* do ambiente dinâmico do sistema de exaustão industrial.

Tabela 18 – Características do vetor observação do ambiente do exaustor industrial.

Ordem	**Observação**	**Limite mínimo**	**Limite máximo**
0	Erro	-∞	∞
1	Integral do Erro	-∞	∞
2	Derivada do Erro	-∞	∞

A função que representa o incentivo para o ambiente dinâmico em análise se assemelha

aos formulados nos problemas de rastreamento. Neste caso a função do incentivo do sistema exaustão sugerida é descrita por

$$I_{fan}(t) = 10(|e| < 0.1) - 5(|e| \geq 0.1) - 100(y \leq 0||y \geq 1000). \tag{53}$$

.

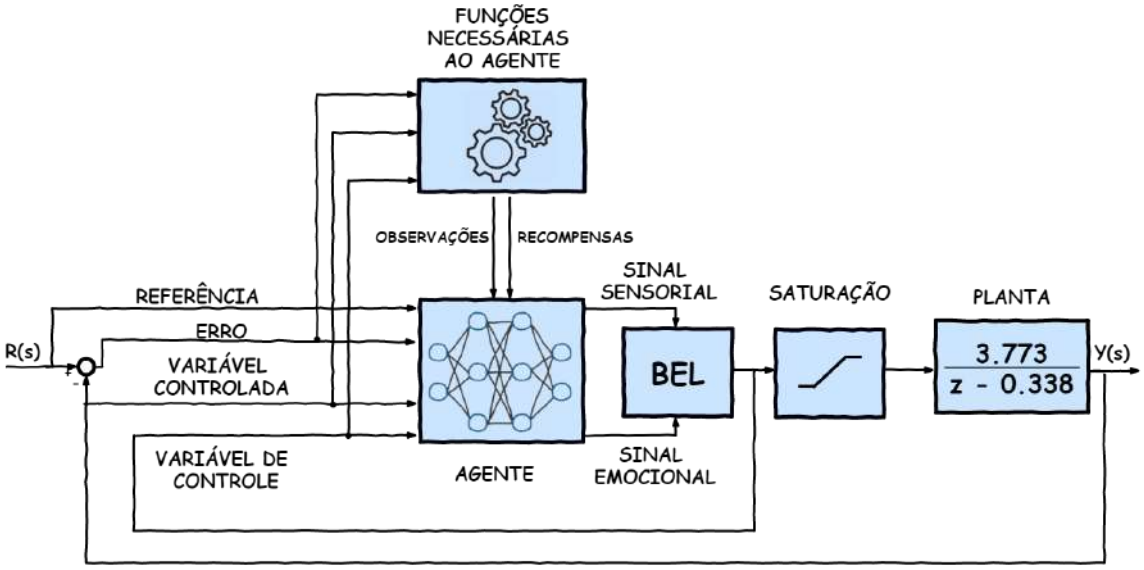
De acordo com (53), o limite de erro para o recebimento da recompensas positivas é 0.1. Por outro lado, caso a variável controlada exceda os limites $y \leq 0$ ou $y \geq 1000$, a recompensa é decrementada e o ambiente é reiniciado.

5.4.1.1 Treinamento no ambiente do exaustor industrial

Uma vez que o problema abordado no ambiente do sistema de exaustão é caracterizado por ser de rastreamento, utiliza-se a mesma metodologia abordada nos ambientes do submarino e braço robótico, anteriormente avaliados neste trabalho.

A Figura 97 apresenta o esquema do sistema de controle do DBELBIC no ambiente do sistema de exaustão industrial.

Figura 97 – Esquema do sistema de controle do sistema de exaustão com DBELBIC utilizado para o treinamento dos estímulos.



Fonte: próprio autor.

De acordo com a Figura 97, nota-se que o agente pode utilizar os *sinais do erro*, *variável controlada*, *variável de controle* e *referência* para para compor os estímulos sensorial e emocional do controlador DBELBIC (arquitetura indireta).

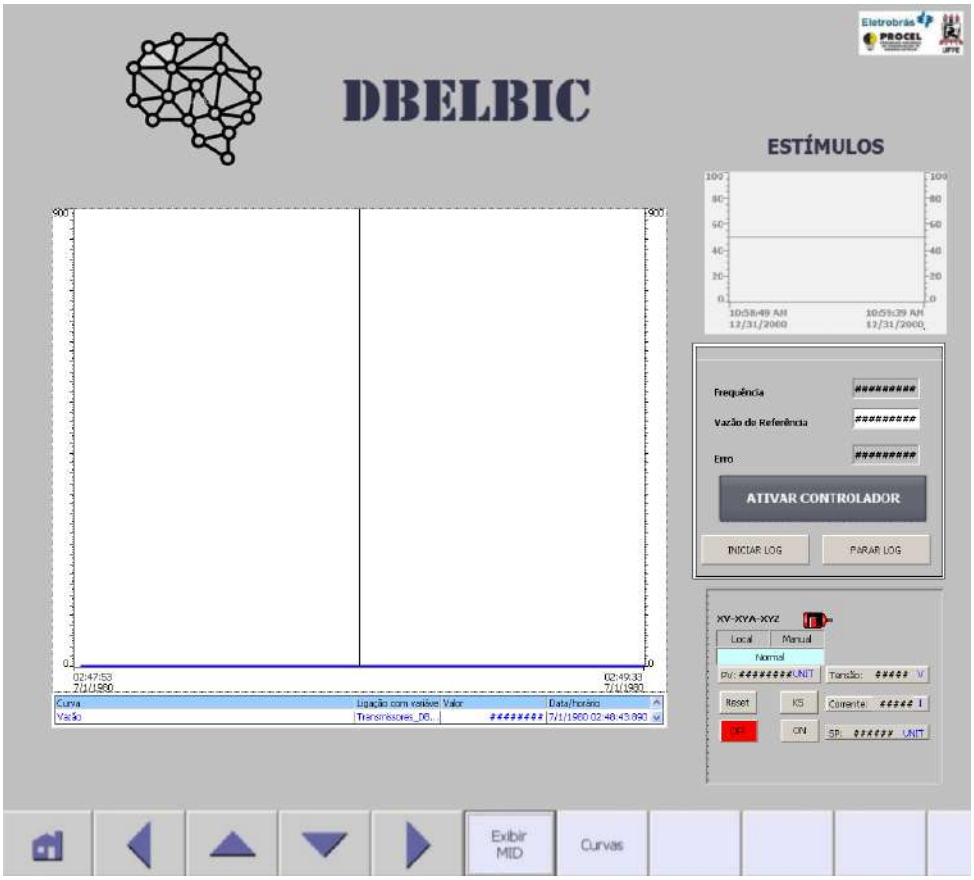
Uma vez que se trata de um sistema real, existem restrições maiores do que nos problemas anteriormente abordados, as quais permitem o funcionamento seguro do processo. Assim sendo,

o sinal de controle proveniente do módulo BEL é limitado por um bloco de saturação em um valor de 30 *Hz* elétricos .

Além disso, optou-se por gerenciar o uso do controlador DBELBIC via sistema supervisório. Neste caso, o usuário precisa permitir que este controlador seja ativado e, consequentemente, entre em operação via *hardware Raspberry Pi*. Por outro lado, caso o sistema perca a comunicação com o controlador DBELBIC no *hardware Raspberry Pi*, o sistema suspende a operação.

A Figura 98 apresenta o supervisório do DBELBIC em *WinCC*®.

Figura 98 – Supervisório do DBELBIC *WinCC*®.

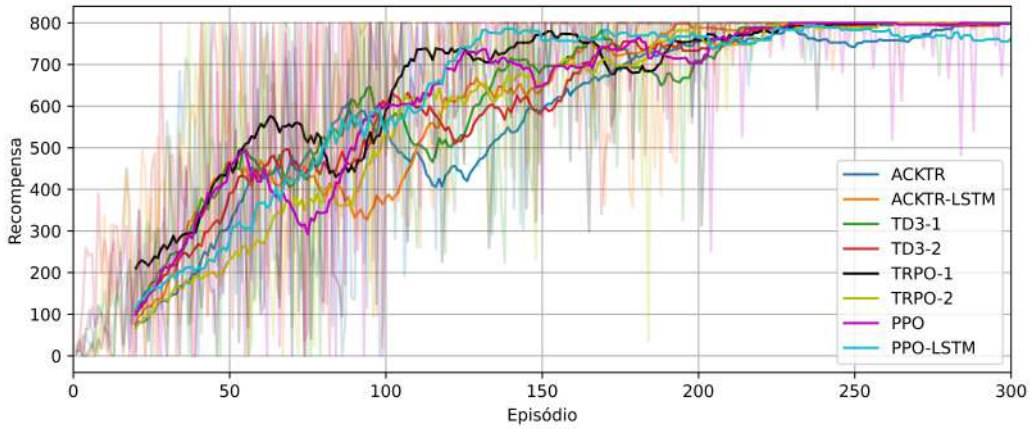


Fonte: próprio autor.

Assim como nos ambientes do submarino e braço robótico, o treinamento no caso do exaustor industrial é realizado via *Simulink*® com o agente de DRL em *Python*. Nesse caso, utilizam-se os mesmos agentes (ACKTR, PPO, TD3 e TRPO). Por outro lado, o mapeamento prévio da estabilidade neste ambiente demonstrou que a referência (r), a variável controlada (y) e a variável de controle (u) não são aconselháveis para compor os estímulos deste controlador. Essa situação está relacionada ao fato de que a presença dessas variáveis restringe bastante a estabilidade do DBELBIC.

Por fim, realiza-se o treinamento do DBELBIC no ambiente do sistema de exaustão industrial do LAMOTRIZ. A Figura 99 apresenta o resultado do treinamento neste ambiente, obtido a partir da utilização de diferentes agentes na produção dos estímulos do DBELBIC.

Figura 99 – Treinamento do DBELBIC com diferentes agentes no ambiente do exaustor.



Fonte: próprio autor.

De acordo com a Figura 99, é visto que todos dos agentes de DRL obtiveram recompensas satisfatórias ao final do período de treinamento. Nesta situação, o critério de parada adotado para o treinamento foi de 300 episódios. Assim sendo, o treinamento durou cerca de *1 hora* e *40 minutos*, uma vez que cada episódio tinha cerca de *20 s* de duração.

A partir do treinamento realizado, optou-se por utilizar apenas o agente PPO-LSTM, uma vez que este agente ainda não foi avaliado nos problemas de rastreamento abordados neste trabalho. Desta forma, após o mapeamento da estabilidade e do treinamento do agente, os sinais de estímulos sensorial e emocional são formulados da seguinte maneira

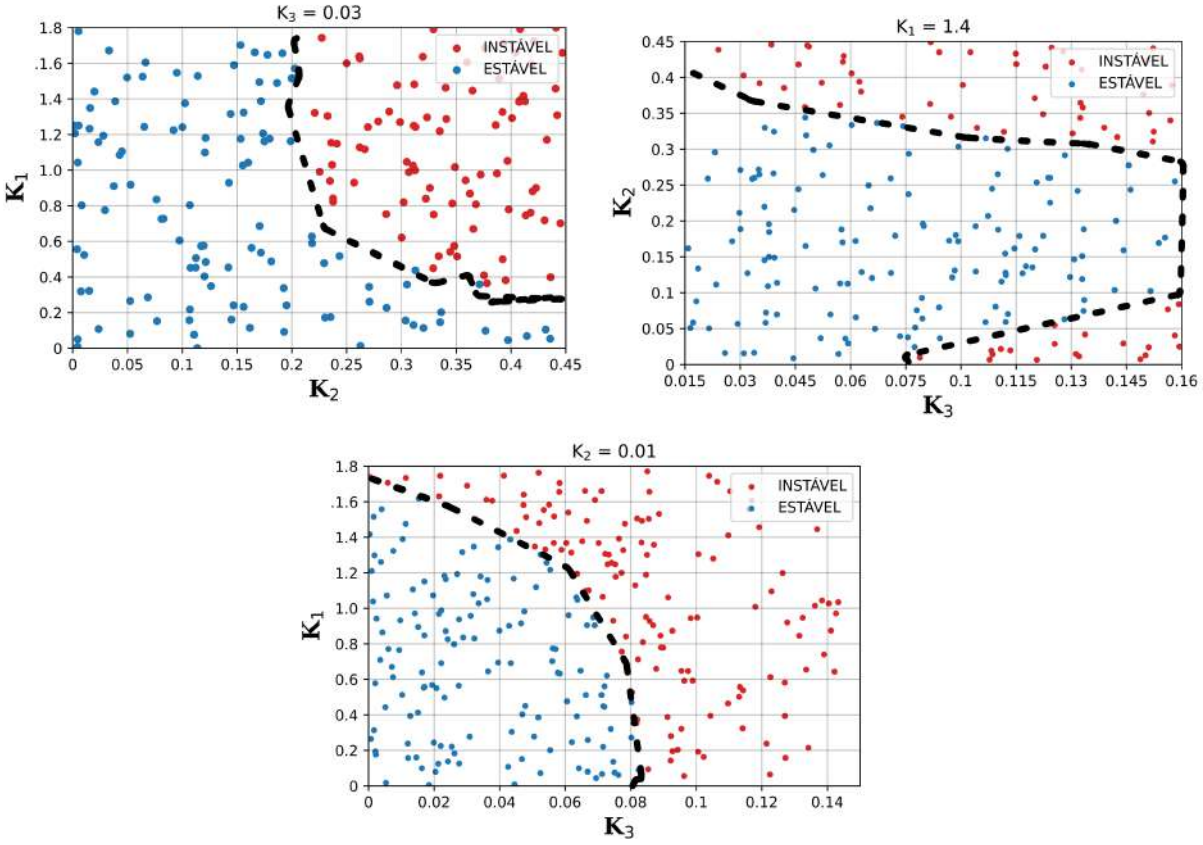
$$S_{fan} = K_1, \tag{54}$$

$$R_{fan} = K_2 e + K_3 \int e, \tag{55}$$

onde K_1, K_2 e K_3 são os ganhos provenientes das DNNs do agente de DRL. Diferentemente dos ambientes dinâmicos do submarino, braço robótico e pêndulo invertido, a única variável da malha de controle presente na definição da arquitetura do DBELBIC é o erro (e). Além disso, a critério de segurança, limitaram-se os ganhos K a uma variação de 10% nos valores obtidos via treinamento *offline*.

De posse dos ganhos *K* nos estímulos *S* e *R* em (54) e (55), respectivamente, apresenta-se na Figura 100 um mapeamento de estabilidade em alguns pontos de operação do controlador no ambiente do exaustor industrial.

Figura 100 – Mapeamento da estabilidade a partir dos ganhos K_1, K_2 e K_3 no ambiente do exaustor industrial.



Fonte: próprio autor.

A partir da Figura 100 é possível perceber as relações entre os ganhos *K* no ambiente do exaustor industrial. Nesse caso, nota-se em especial que o ganho K_3 associado à variável da integral do erro no estímulo *R* é o que apresenta maior destaque na perda da estabilidade do sistema de controle.

5.4.1.2 Resultados no ambiente do exaustor industrial

De modo realizar uma avaliação de desempenho do DBELBIC neste sistema dinâmico, busca-se compará-lo a um PID tradicional e um FLC, ambos implementados previamente neste sistema dinâmico. No que diz respeito aos parâmetros do PID e do FLC, optou-se por utilizar os valores otimizados a partir de algoritmos metaheurísticos conhecidos como otimização por enxame de partículas - *particle swarm optimization* (PSO) e algoritmos genéticos - genetic algorithm (GA) (SILVA, 2015; SILVA et al., 2019).

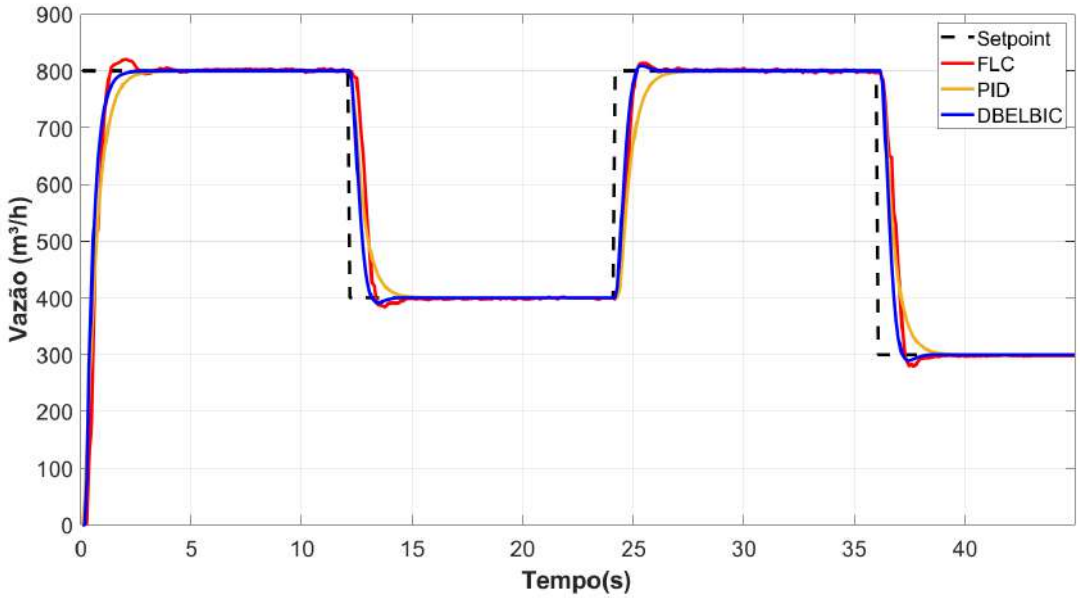
A Tabela 19 apresenta os valores dos parâmetros concernentes ao PID utilizado. Por outro lado, os parâmetros utilizados no FLC podem ser consultados no trabalho (SILVA et al., 2019).

Tabela 19 – Parâmetros do controlador PID utilizado no sistema de exaustão industrial do LAMOTRIZ.

PID		
K_p	K_i	K_d
0.00132	0.02634	0.00000

A Figura 101 apresenta o seguimento de referência dos controladores PID, FLC e DBELBIC juntos ao sistema de exaustão industrial via ambiente em *Simulink*®.

Figura 101 – Desempenho dos controladores PID, FLC e DBELBIC no rastreamento da referência no sistema de exaustão industrial em *Simulink*®.



Fonte: próprio autor.

A partir da Figura 101, é possível notar que os controlados PID, FLC e DBELBIC apresentam desempenhos satisfatórios no rastreamento da referência neste ambiente.

As Tabelas 20[15] e 21 apresentam o desempenho dos controladores em análise neste ambiente.

[15] Análise do primeiro degrau (0 m^3/h - 800 m^3/h).

Tabela 20 – Características dinâmicas das respostas dos controladores PID, FLC e DBELBIC junto ao sistema de exaustão industrial em *Simulink*®.

Controlador	Sobressinal (%)	Tempo de subida (s)	Tempo de acomodação (s)	Erro estacionário (%)
PID	0.08	1.32	3.02	0.00
FLC	3.07	1.11	3.77	0.00
DBELBIC	0.49	1.16	2.72	0.00

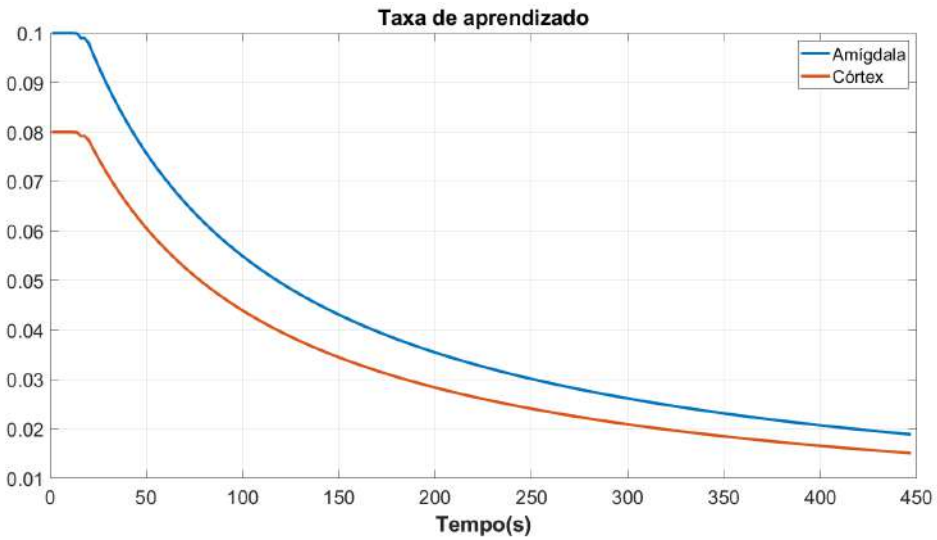
Tabela 21 – Índices de desempenho dos controladores PID, FLC e DBELBIC junto ao sistema de exaustão industrial em *Simulink*®.

Controlador	ISE	IAE	ITAE
PID	$5.06\ 10^6$	$1.42\ 10^4$	$2.25\ 10^6$
FLC	$4.94\ 10^6$	$1.33\ 10^4$	$2.01\ 10^6$
DBELBIC	$4.89\ 10^6$	$1.18\ 10^4$	$1.89\ 10^6$

A partir das Tabelas 20 e 21 é possível observar que em relação aos controladores PID e FLC, o DBELBIC apresentou um melhor desempenho na resposta dinâmica na maioria dos índices avaliados. Além disso, nota-se um desempenho muito similar do DBELBIC ao controlador FLC otimizado via metaheurísticas, o que representa um desempenho muito satisfatório.

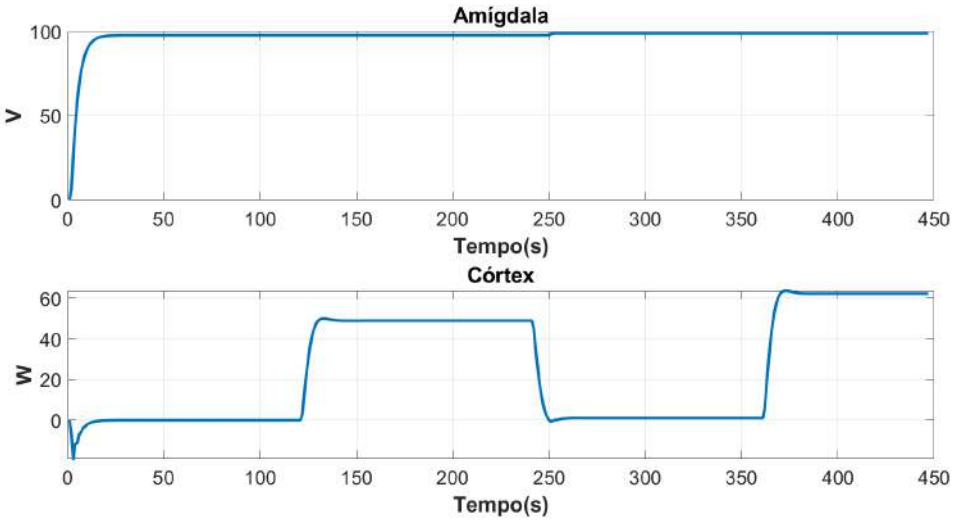
As Figuras 102, 103 e 104 apresentam as taxas de aprendizados α e β, curvas de aprendizados da amígdala e córtex e os sinais do DBELBIC nesta etapa, respectivamente.

Figura 102 – Taxas de aprendizado α e β do DBELBIC junto ao sistema de exaustão industrial em *Simulink*®.



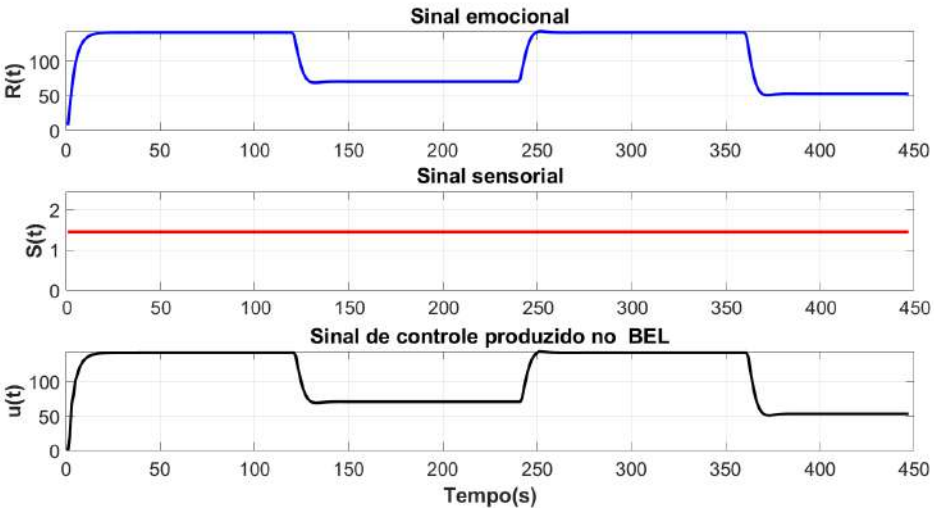
Fonte: próprio autor.

Figura 103 – Curvas de aprendizado do DBELBIC junto ao sistema de exaustão industrial em *Simulink*®.



Fonte: próprio autor.

Figura 104 – Sinais do DBELBIC após o treinamento do agente junto ao sistema de exaustão industrial em *Simulink*®.



Fonte: próprio autor.

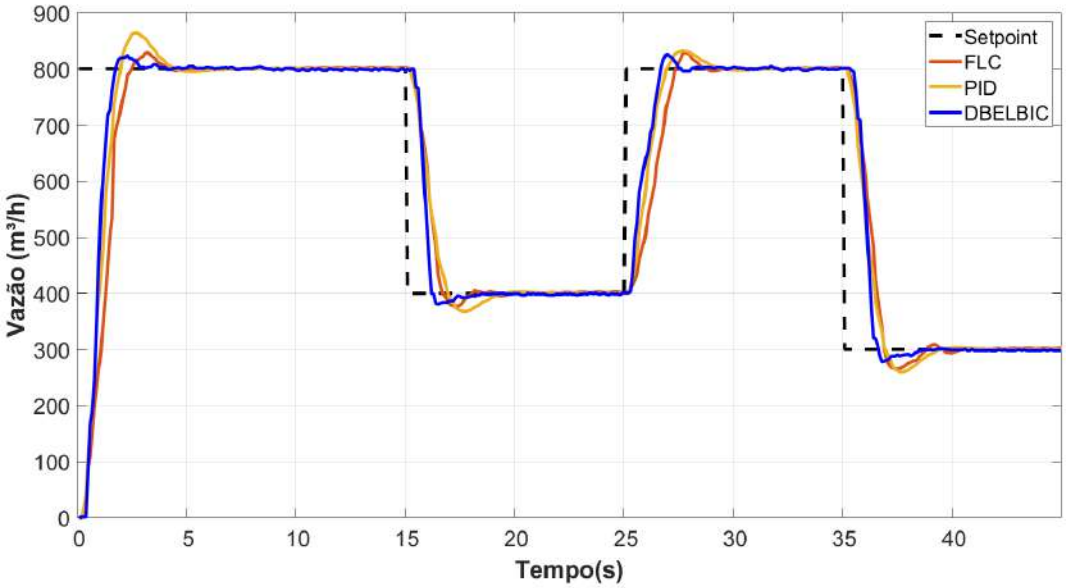
A partir da Figura 104 é possível notar que o sinal sensorial (S) é praticamente constante em todo o funcionamento do DBELBIC, uma vez que este estímulo não se utiliza de nenhuma variável da malha de controle para compor tal sinal. Desta forma, aparentemente todo o sinal emocional (R) que chega ao módulo BEL é gerado como sinal final de controle (u) deste controlador.

No caso deste ambiente dinâmico, a análise principal do DBELBIC se concentra em sua implementação no ambiente real. Assim sendo, algumas análises anteriormente abordadas nos problemas de rastreamento não são relacionadas.

Uma vez finalizadas as etapas de treinamento e simulação do DBELBIC via ambiente *Simulink*®, verifica-se toda a comunicação do *hardware Raspberry Pi* e CLP via Snap7. A partir disso, é feita a aplicação do DBELBIC ao ambiente dinâmico real do exaustor industrial do LAMOTRIZ. Os controladores utilizados a título de comparação com o DBELBIC neste ambiente (PID e FLC) já estão previamente implementados no próprio CLP Siemens® via código *LADDER*[16].

A Figura 105 apresenta o seguimento da referência dos controladores PID, FLC e DBELBIC no ambiente do exaustão industrial do LAMOTRIZ.

Figura 105 – Desempenho dos controladores PID, FLC e DBELBIC no rastreamento da referência no sistema de exaustão industrial no LAMOTRIZ.



Fonte: próprio autor.

De acordo com a Figura 105, percebe-se que os controladores PID, FLC e DBELBIC apresentaram bons desempenhos no seguimento de referência em termos de erro estacionário e resposta transitória. Todavia, referindo-se ao DBELBIC foi necessário realizar um pequeno ajuste de escala na saída do controlador, uma vez que não houve um treinamento *online*.

As Tabelas 22[17] e 23 apresentam o desempenho dos controladores em análise neste ambiente.

[16] Linguagem de alto nível utilizada na programação CLPs.
[17] Análise do primeiro degrau (0 m^3/h - 800 m^3/h).

Tabela 22 – Características dinâmicas das respostas dos controladores PID, FLC e DBELBIC junto ao sistema de exaustão industrial real.

Controlador	Sobressinal (%)	Tempo de subida (s)	Tempo de acomodação (s)	Erro estacionário (%)
PID	8.02	1.89	4.22	0.35
FLC	3.68	1.58	3.90	0.27
DBELBIC	3.73	1.42	3.81	0.54

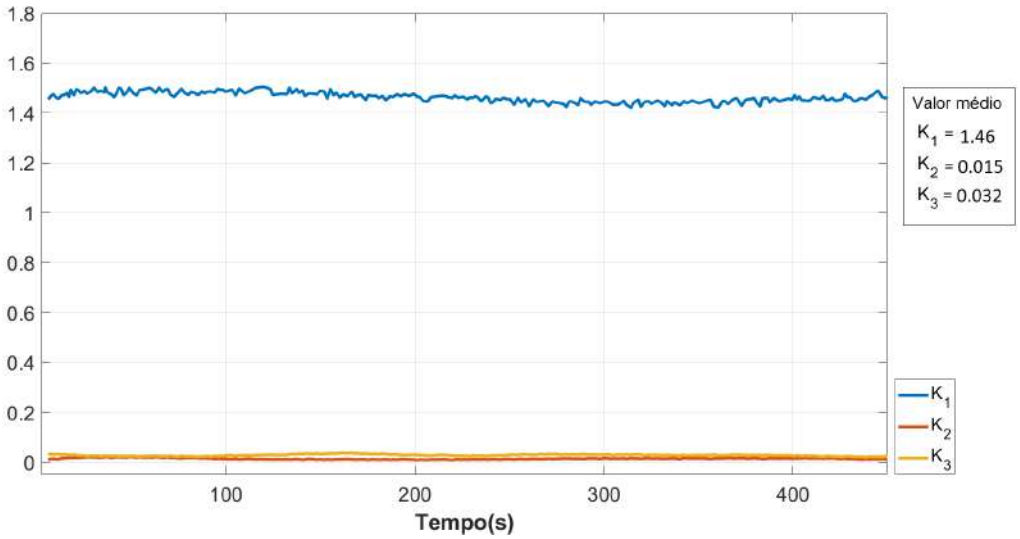
Tabela 23 – Índices de desempenho dos controladores PID, FLC e DBELBIC junto ao sistema de exaustão industrial real.

Controlador	ISE	IAE	ITAE
PID	$1.05\ 10^7$	$2.56\ 10^4$	$2.63\ 10^8$
FLC	$8.72\ 10^6$	$2.23\ 10^4$	$2.37\ 10^8$
DBELBIC	$8.41\ 10^6$	$1.96\ 10^4$	$2.27\ 10^8$

A partir da Tabela 22, nota-se um maior erro estacionário por parte do DBELBIC, sendo este um dos principais critérios para a escolha de um controlador, tal fato apresenta relevância nos resultados. No entanto, pode-se justificar tal situação pelo fato do sistema não ter sido treinado online na planta em questão, afetando assim o desempenho do controlador proposto.

A a Figura 106 apresenta o comportamento dos ganhos K, definidos em (54) e (55), respectivamente, referindo-se ao seguimento da referência descrito na Figura 105.

Figura 106 – Ganhos nos estímulos do DBELBIC no seguimento de referência no ambiente do exaustor industrial.



Fonte: próprio autor.

Apesar do modelo do sistema de exaustão industrial não apresentar um grau de complexidade muito grande (sistema de primeira ordem), a aplicabilidade do controlador DBELBIC neste caso serviu como um bom parâmetro avaliativo de seu desempenho frente a sistemas reais. De forma geral, o desempenho apresentado foi satisfatório, no entanto, existe a possibilidade de melhorá-lo a partir de um treinamento *online* na planta.

5.5 Considerações finais

A concepção e desenvolvimento de controladores, em especial na área da AI, caracteriza-se por ser uma tarefa árdua que exige diversos testes e um exaustivo comissionamento. Nesse sentido, utilizar-se de controladores com aplicação comprovada como referencial construtivo é essencial para atestar o funcionamento de uma proposta.

Neste trabalho, a partir da concepção das arquiteturas de estímulos direta e indireta no DBELBIC, foi possível demonstrar a aplicabilidade deste tipo de controlador em diferentes classes de problemas, recorrentes na engenharia de controle. A partir do uso de controladores distintos (PID, FLC, BELBIC e G-BELBIC) em diferentes ambientes dinâmicos, foi possível comparar o desempenho do DBELBIC nas mesmas condições dinâmicas.

Apesar dos desafios inerentes ao dimensionamento, treinamento dos agentes de DRL e o consequente *deployment* para utilização prática da proposta do controlador DBELBIC, os resultados demonstraram a ampla capacidade deste controlador em atingir desempenhos satisfatórios em ambientes dinâmicos diversos e, até mesmo superar outros controladores já utilizados.

No que se refere às limitações da proposta, destaca-se o uso do *hardware Raspberry Pi* que inviabilizou o treino *online*. Uma vez que o esforço computacional dedicado ao treinamento das DNNs é relativamente considerável, não foi possível continuar o treinamento do controlador embarcado. Todavia, uma vez treinado o sistema, pode-se facilmente utilizá-lo em um *hardware* com baixo processamento. Além disso, a proposta do DBELBIC é limitada à utilização do *framework TensorFlow* para a concepção dos agentes de DRL, o que restringe sua viabilidade para outras linguagens de programação. No que diz respeito ao uso de DNNs para formar o controlador DBELBIC, verifica-se uma limitação na flexibilidade de mudanças na estrutura dos estímulos, caso haja a necessidade de melhorar um desempenho, deve-se continuar o treinamento. Por fim, a característica construtiva do DBELBIC restringiu bastante alguns aspectos inerentes a sua análise a partir das técnicas tradicionais da engenharia de controle.

6 CONCLUSÕES E TRABALHOS FUTUROS

O controlador emocional representa uma classe importante de controladores baseados nas características de aprendizado dos seres vivos. Este tipo de controlador tem por objetivo emular características específicas da aprendizagem com base nas emoções, aplicando-as em sistemas de controle dinâmicos (LOTFI; REZAEE, 2018; HAITH; W., 2013).

De modo geral, as aplicações do controlador emocional apresentam bons resultados no controle de sistemas dinâmicos, principalmente no que se refere à sua velocidade de resposta (LUCAS; SHAHMIRZADI; SHEIKHOLESLAMI, 2004; SHARBAFI; LUCAS; DANESHVAR, 2010). O desempenho deste controlador está associado diretamente à formulação dos sinais de estímulos emocional e sensorial que constituem seu modelo. Este fato pode representar um grande desafio em sua concepção, uma vez que tais sinais não apresentam uma regra definida e, geralmente, podem variar significativamente para cada sistema dinâmico em questão (SADEGHI; DARYABEIGI, 2014; DEHKORDI et al., 2011; MARKADEH et al., 2011).

Neste contexto, o trabalho apresentou uma metodologia para a determinação generalista da arquitetura dos sinais de estímulos emocional e sensorial do controlador emocional, denominando-se DBELBIC. A partir da utilização do estado da arte das técnicas de DRL, tornou-se possível adaptar as necessidades da formulação dos estímulos do controlador emocional à capacidade adaptativa de tais técnicas.

A associação do RL com as recentes técnicas desenvolvidas na área da DL permitiram obter algoritmos que apresentam melhores desempenhos do que em anos anteriores (SUTTON; BARTO, 2018; GOODFELLOW; BENGIO; COURVILLE, 2016; GOODFELLOW et al., 2014). A partir do uso desses recursos na elaboração dos estímulos do controlador emocional, obteve-se como resultado um controlador com boas características dinâmicas sem requere uma grande expertise de um especialista. Por outro lado, a formulação das recompensas para os agentes de DRL continuou a ser um parâmetro empírico.

De modo a permitir a caracterização dos estímulos do controlador emocional por meio das técnicas de DRL, propuseram-se arquiteturas distintas. A partir da utilização de DNNs como parâmetros adaptativos dos estímulos emocional e sensorial, as arquiteturas direta e indireta foram estabelecidas. A arquitetura direta dos sinais faz uso diretamente da saída da DNN para criar os sinais de estímulos. Por outro lado, a arquitetura indireta utiliza a saída da DNN para controlar os ganhos associados as leis de controle que compõem os estímulos, os quais se utilizam de sinais da malha de controle.

Uma vez determinada a arquitetura a ser utilizada nos estímulos do controlador emocional, fez-se necessário estabelecer os algoritmos mais adequados para a tarefa proposta. Neste sentido, optou-se por utilizar os agentes de DRL presentes na biblioteca *OpenAI Baselines* (BROCKMAN

et al., 2016). A utilização desta biblioteca permitiu implementar o estado da arte dos agentes de DRL de forma prática e mais ágil, uma vez que o código fonte está em constante atualização por parte dos desenvolvedores.

Os agentes utilizados neste trabalho a partir da biblioteca *OpenAI Baselines* foram baseados no *framework Tensorflow*. Os agentes possuem um modelo AC para extração das informações do ambiente dinâmico e, consequentemente, tomada de decisão. A partir de diferentes possibilidades de modelos de DNNs, optou-se por utilizar o tipo *Feedfoward* e *LSTM*, justificando-se pela natureza das informações coletadas do ambiente dinâmico (erro em regime, variáveis controladas, variáveis de controle, etc.).

Além de apresentar uma proposta de metodologia para a determinação dos estímulos do controlador emocional por meio das técnicas de DRL, o trabalho propôs uma nova abordagem para o aprendizado dos estímulos. A partir de resultados de testes e simulações, verificou-se que utilizar uma taxa de aprendizado com um decaimento exponencial para ambos os modelos, amígdala e córtex, torna possível oferecer velocidades de aprendizado diferenciadas para diferentes momentos de atuação do controlador e, desta maneira ter resultados mais estáveis para o sinal de controle. Neste sentido, a adoção deste modelo de taxa da aprendizado, variável com o tempo, representa uma alternativa eficaz ao modelo atual proposto pelo BELBIC, uma vez que valores constantes de taxas de aprendizados podem estagnar o aprendizado deste controlador.

No geral, a proposta deste trabalho envolveu etapas de desenvolvimento, simulações e implementações práticas do controlador. Por esta razão, várias ferramentas foram necessárias para a elaboração, treinamento, testes e aplicação prática do DBELBIC. Neste sentido, fez-se necessária a adoção de plataformas distintas (*MATLAB*® e *Python*) para o desenvolvimento do trabalho, enfatizando-se o intercâmbio de informações, fundamental para a metodologia desta proposta. Por outro lado, referindo-se à implementação prática do controlador, optou-se por utilizar o *hardware Raspberry Pi*, uma vez que possui um baixo custo de aquisição e permite uma grande flexibilidade de aplicação, proporcionada principalmente por sua grande quantidade recursos disponíveis (RASPBERRY, 2019; BAUERMEISTER, 2019).

De forma a realizar a avaliação do DBELBIC, utilizou-se a classe de problemas do rastreamento da referência e da regulação, comparando-se o controlador proposto a outros tipos de controladores comuns na engenharia de controle (OGATA, 2003). Cada um desses problemas envolvia um sistema dinâmico distinto, tal que permitissem demonstrar a eficácia da proposta em obter adequadamente os estímulos sensorial e emocional do DBELBIC. Dessa forma, permitiu-se atingir bons níveis de desempenho dinâmico para este tipo de controlador emocional. Neste caso, os treinamentos e simulações utilizaram os recursos do *software MATLAB*® para modelar o ambiente dinâmico. O agente, por sua vez, foi formulado em *Python*, utilizando-se dos recursos das bibliotecas de DRL. Por fim, uma vez que o objetivo da proposta de controlador foi sua aplicação em sistemas embarcados, a partir de ensaios e simulações, a comunicação entre o agente e o ambiente foi definida como serial.

Em resumo, os resultados apresentados durante o trabalho demonstraram a eficácia da proposta de utilização das técnicas de DRL para a obtenção das leis que regem os estímulos sensorial e emocional do controlador DBELBIC. Tal desempenho dinâmico se refere, principalmente, à velocidade da resposta, minimização do erro em regime e estabilidade do processo controlado.

De forma a corroborar com a proposta deste trabalho, fez-se necessária a avaliação prática deste tipo de controlador em um processo dinâmico real, avaliando assim os aspectos da viabilidade da implementação em processos industriais. Apesar do sistema utilizado nesta avaliação (exaustor industrial) apresentar dinâmicas de primeira ordem, permitiu-se avaliar diversos parâmetros práticos relevantes no processo de implementação deste controlador. Neste sentido, uma questão importante foi a comunicação entre o controlador e o processo dinâmico. Assim sendo, após diversas avaliações de possíveis alternativas, optou-se por utilizar o pacote de comunicação Snap7. O Snap7 permitiu realizar uma comunicação serial simples e direta entre o agente implementado no *hardware Raspberry Pi* e um CLP *Siemens*® *S*7-300.

No que diz respeito à implementação prática do DBELBIC, fez-se necessário adicionar cuidados adicionais visando a segurança final do processo controlado, a exemplo da limitação de variação paramétrica dos ganhos nos estímulos. Além disso, diversos desafios ainda são pertinentes ao modelo da proposta deste trabalho, uma vez que a utilização das técnicas de DRL ainda representam um desafio prático por parte dos projetistas (LILLICRAP et al., 2016).

Por fim, ainda é necessária avaliações em geral do controlador proposto, principalmente no que se refere ao tipo de problema envolvido e, questões importantes como a estabilidade. Apesar dos desafios que representam a metodologia e utilização deste tipo de controlador, os resultados apresentados neste trabalho demonstraram que a utilização das técnicas de DRL, associadas ao controlador emocional, podem contribuir na melhora de desempenho deste tipo de controlador, permitindo assim obter uma maior faixa de aplicação em sistemas dinâmicos distintos.

6.1 Trabalhos futuros

O trabalho apresentou uma ampla possibilidade para o desenvolvimento de futuras pesquisas no que diz respeito à aplicação das técnicas de DRL em sistemas de controle. Por outro lado, questões referentes à possibilidade de aprimorar-se controladores baseados em sistemas de inteligência artificial também podem ser exploradas.

No caso controlador emocional, identificou-se durante a pesquisa que abordagens distintas, a partir do uso de diferentes técnicas de inteligência artificial podem ser utilizadas em posteriores trabalhos. Uma vez que o modelo de aprendizado neste controlador é simplificado em diversos níveis (SHAHMIRZADI, 2005), equações adicionais podem ser formuladas para representar o tálamo, hipocampo, hipotálamo, entre outros (LEDOUX; FELLOUS, 1995;

OHMAN; MINEKA, 2001; MORéN, 2002). Além disso, pode-se fazer uso do modelo BEL associado a técnicas distintas de controle, fornecendo novos modelos híbridos para este tipo de controlador.

A metodologia do trabalho ficou restrita ao uso do *framework Tensorflow*. A motivação disto está relacionada a razões de implementação prática e disponibilidade de agentes na biblioteca *OpenAI Baselines*, o que porventura limitou alguns aspectos avaliativos. Uma alternativa a esta limitação é a possibilidade de utilizar outros *frameworks*, como por exemplo o *framework Pytorch* (CHOLLET, 2018; FAIR, 2016). O *Pytorch* possui uma grande comunidade *online* e vem ganhando cada vez mais espaço em diversas aplicações. A razão disto está associada à forma de como foi desenvolvido, programado para ser mais rápido, linear e intuitivo do que o próprio *Tensorflow* e, além disso, possui uma API muito mais simples e amigável (FAIR, 2016).

Os agentes de DRL utilizados neste trabalho fizeram uso das arquiteturas do tipo *Feedfoward* e *LSTM*. Assim sendo, propõe-se utilizar novas arquiteturas e até diferentes métodos para a obtenção de hiperparâmetros, visando atingir desempenhos melhores que os verificados até então na presente proposta. Neste sentido, pode-se utilizar, como alternativa, técnicas de *AUTOML*[1] como o *feature engineering, meta learning, architecture search* e *hyperparameter optimization* (HE; ZHAO; CHU, 2020). Por meio das técnicas de AUTOML é possível obter de forma otimizada as arquiteturas das DNNs.

A partir da infraestrutura utilizada no trabalho, o ajuste fino do controlador implementado não se fez possível, devido principalmente a questões de limitação do *hardware Raspberry Pi*. Por esta razão, pode-se optar em futuros trabalhos na utilização de outro *hardware* com maior capacidade de processamento. Neste caso, é possível realizar treinamentos dos estímulos do controlador proposto de forma *online*, realizando assim um ajuste fino do controlador em questão. Por outro lado, a utilização do *Raspberry Pi* permite a possibilidade de adoção da *IoT* para o monitoramento, armazenamento e gerenciamento do sistema de controle em questão.

Em suma, diversos são os desafios e oportunidades referentes aos temas relacionados a este trabalho. Constantes mudanças e avanços nas técnicas de controladores inteligentes, bem como nos modelos de sistemas de inteligência artificial, permitem cada vez mais obter melhores desempenhos e, consequentemente, acarretam desafios cada vez mais complexos.

[1] Abreviação do inglês Automated Machine Learning ou Aprendizado de Máquina Automatizado.

REFERÊNCIAS

3D4MEDICAL. *O Sistema Límbico.* 2019. Disponível em: <https://3d4medical.com/blog>. Acesso em: 28 jul. 2019. Citado na página 71.

ACADEMY, D. S. *Uma breve história das redes neurais artificiais* . 2019. Disponível em: <http://deeplearningbook.com.br/uma-breve-historia-das-redes-neurais-artificiais/>. Acesso em: 01 agos. 2019. Citado na página 54.

ACHUTTID, A. *Circuito de recompensas no cérebro.* 2010. Disponível em: <http://achutti.blogspot.com>. Acesso em: 25 jul. 2019. Citado na página 67.

ADAM. *Hipotálamo.* 2018. Disponível em: <http://aia5.adam.com/content.aspx?productid=118&pid=6&gid=19239>. Acesso em: 14 jul. 2019. Citado 2 vezes nas páginas 76 e 77.

AFIFI, A. K.; BERGMAN, R. A. *Functional Neuroanatomy: Text and Atlas.* 1. ed. [S.l.]: McGraw-Hill Professional., 1997. Citado na página 71.

AGGLETON, J. P. *The amygdala: Neurobiological aspects of emotion, memory, and mental dysfunction.* 1. ed. NY, US: Wiley-Liss., 1992. Citado 2 vezes nas páginas 77 e 78.

AKKAYA, I. et al. Solving rubik's cube with a robot hand. *arXiv:1910.07113 [cs.LG]*, 2019. Citado na página 65.

ALCÂNTARA, A. P.; ALCÂNTARA, D. S. Otimização do erro de deslocamento de um braço robótico com 1 gdl aplicando cinemática direta. *Mostra Nacional de Robótica (MNR)* ., 2013. Citado na página 141.

ANDERSON, C. C. et al. Synthesis of reinforcement learning, neural networks, and pi control applied to a simulated heating coil. *Journal of Artificial Intelligence in Engineering*, v. 11, p. 423–431, 1997. Citado na página 53.

ANTONIO, V. E. et al. Neurobiology of the emotions. *Archives of Clinical Psychiatry (São Paulo), SciELO Brasil* ., v. 35, p. 55–65, 2008. Citado na página 69.

ARBIB, M. A. *The Handbook of Brain Theory and Neural Networks.* 2. ed. MA, USA: MIT Press., 2002. Citado na página 75.

ARMONY, J. L.; LEDOUX, J. E. How the brain processes emotional information. *Annals of the New York Academy of Sciences: Psychobiology of posttraumatic stress disorder*, v. 82, p. 259–270, 1997. Citado 2 vezes nas páginas 73 e 75.

ARULKUMARAN, K. et al. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine.*, v. 34, p. 26–38, 2017. Citado 5 vezes nas páginas 29, 55, 65, 114 e 129.

ATLAS psychology. *Amígdala.* 2017. Disponível em: <https://quantum-psychology-atlas/intuition-and-the-resonances-of-orbitofrontal-cortex>. Acesso em: 10 jul. 2019. Citado na página 74.

BABAIE, T.; KARIMIZANDI, R.; LUCAS, C. Learning based brain emotional intelligence as a new aspect for development of an alarm system. *Soft Computing.*, v. 9, p. 857–873, 2008. Citado na página 84.

BAIRD, L. Residual algorithms: Reinforcement learning with function approximation. *Proceedings of the International Conference on Machine Learning.*, p. 30–37, 1995. Citado na página 27.

BAKKER, I. D. *State of open source deep learning frameworks in.* 2018. Disponível em: <https://towardsdatascience.com/battle-of-the-deep-learning-frameworks-part-i-cff0e3841750>. Acesso em: 14 jul. 2019. Citado na página 85.

BARD, P. A diencephalic mechanism for the expression of rage with special reference to the sympathetic nervous system. *American Journal of Physiology.*, v. 84, p. 490–516, 1928. Citado na página 69.

BARRETO, A. et al. Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences Dec 2020, DOI: 10.1073/pnas.1907370117*, v. 117, p. 30079–30087, 2020. Citado na página 26.

BARTO, A. G.; SUTTON, R. S.; ANDERSON, C. W. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics.*, v. 13, p. 835–846, 1983. Citado 4 vezes nas páginas 26, 28, 152 e 153.

BAUERMEISTER, G. *Nova Raspberry Pi 3 B+*. 2019. Disponível em: <https://blog.fazedores.com/nova-raspberry-pi-3-b-plus/>. Citado 4 vezes nas páginas 85, 87, 168 e 182.

BEHESHTI, Z.; HASHIM, S. Z. M. A review of emotional learning and it's utilization in control engineering. *Int. J. Advance. Soft Comput.*, 2010. Citado 3 vezes nas páginas 69, 73 e 80.

BEKLEMYSHEVA, A. *Why Use Python for AI and Machine Learning?* 2015. Disponível em: <https://steelkiwi.com/blog/python-for-ai-and-machine-learning/>. Citado na página 84.

BELLMAN, R. E. *Dynamic programing.* 1. ed. NJ, USA: Princeton University Press, 1957. Citado 2 vezes nas páginas 25 e 46.

BELLMAN, R. E. Dynamic programming and stochastic control processes. *Information and Control.*, v. 1, p. 228–239, 1958. Citado na página 25.

BELRESEARCH. *Amígdala.* 2017. Disponível em: <http://belresearch.org/>. Acesso em: 08 jul. 2019. Citado na página 72.

BENGIO, Y.; COURVILLE, A.; VINCENT, P. Representation learning: A review and new perspectives. *IEEE Trans. on Pattern Analysis and Machine Intelligence.*, v. 8, p. 1798–1828, 2013. Citado na página 29.

BERKENKAMP, F. et al. Safe model-based reinforce- ment learning with stability guarantees. *Advances in Neural Information Processing Systems.*, p. 908–919, 2017. Citado na página 110.

BERTSEKAS, D. P. *Dynamic Programming: Deterministic and Stochastic Models.* 1. ed. NJ, USA: Prentice-Hall, 1987. Citado na página 26.

BOBITI, R.; LAZAR, M. A sampling approach to

nding lyapunov functions for nonlinear discrete-time systems. *Proc. of the IEEE European Control Conference.*, v. 3, p. 561–566, 2016. Citado na página 109.

BRADTKE, S. J.; BARTO, A. G. Linear least-squares algorithms for temporal difference learning. *Machine Learning.*, v. 22, p. 33–57, 1996. Citado 2 vezes nas páginas 27 e 65.

BRADTKE, S. J.; YDSTIE, B. E.; BARTO, A. G. Adaptive linear quadratic control using policy iteration. *Proceedings of the American Control Conference.*, p. 3475–3479, 1994. Citado na página 65.

BROCKMAN, G. et al. Openai gym. *arXiv:1606.01540v1.*, v. 1, 2016. Citado 6 vezes nas páginas 90, 93, 94, 102, 123 e 182.

BROMBERG-MARTIN, E. S. et al. A pallidus-habenuladopamine pathway signals inferred stimulus values. *Journal of Neurophysiology.*, v. 104, p. 1068– 1076, 2010. Citado na página 31.

BUSONIU, L. et al. *Reinforcement Learning and Dynamic Programming Using Function Approximators.* 1. ed. FL, USA: CRC Press, 2010. Citado 3 vezes nas páginas 25, 26 e 27.

CANNON, W. B. The james-lange theory of emotions: A critical examination and an alternative theory. *The American Journal of Psychology.*, v. 39, p. 106–124, 1927. Citado na página 69.

CANNON, W. B. Again the james-lange and the thalamic theories of emotion. *Psychological Review.*, v. 38, p. 281–295, 1931. Citado na página 69.

CESAR, M. B. et al. Brain emotional learning based control of a sdof structural system with a mr damper. *CONTROLO 2016. Lecture Notes in Electrical Engineering*, v. 402, p. 547–557, 2017. Citado na página 31.

CHOLLET, F. *Deep Learning with Python.* 1. ed. [S.l.]: Manning Publications, 2018. Citado na página 184.

DALGLEISH, T. Nature reviews neuroscience. *The emotional brain.*, v. 5, p. 583–589, 2004. Citado na página 73.

DASHTI, Z.; GHOLAMI, M.; HAJIMANI, M. Brain emotional learning based intelligent controller for velocity control of an electro hydraulic servo system. *IOSR J. Electr. Electron. Eng.*, v. 12, p. 29–35, 2017. Citado na página 109.

DAW, N. D.; SHOHAMY, D. The cognitive neuroscience of motivation and learnig. *Social Cognition.*, v. 26, p. 593–620, 2008. Citado na página 31.

DEHKORDI, B. M. et al. A comparative study of various intelligent based controllers for speed control of ipmsm drives in the field-weakening region. *Expert Systems with Applications.*, v. 38, p. 12643–12653, 2011. Citado 3 vezes nas páginas 33, 81 e 181.

DENG, L. A tutorial survey of architectures, algorithms, and applications for deep learning. *APSIPA Transactions on Signal and Information Processing.*, p. 1–29, 2014. Citado na página 28.

DHARIWAL, P. et al. *OpenAI Baselines*. 2016. Disponível em: <https://github.com/openai/baselines>. Acesso em: 17 jul. 2019. Citado 2 vezes nas páginas 123 e 154.

DOLAN, R. J.; DAYAN, P. Goals and habits in the brain. *Neuron.*, p. 312–325, 2013. Citado na página 65.

DOLL, B. B.; SIMON, D. A.; DAW, N. D. The ubiquity of model-based reinforcement learning. *Current Opinion in Neurobiology.*, v. 22, p. 1–7, 2012. Citado na página 31.

DORF, R. C.; BISHOP, R. H. *Sistemas de Controle Moderno.* 13. ed. Rio de Janeiro, Brasil: LTC, 2018. Citado 4 vezes nas páginas 25, 108, 119 e 152.

DULAC-ARNOLD, G.; MANKOWITZ; HESTER, T. Challenges of real-world reinforcement learning. *arXiv:1904.12901 [cs.LG]*, 2019. Citado na página 27.

EICKHOLD, J. *Serial Communication in Java with Raspberry Pi and RXTX.* 2012. Disponível em: <https://eclipsesource.com/blogs/2012/10/17/serial-communication-in-java-with-raspberry-pi-and-rxtx/>. Acesso em: 26 fev. 2020. Citado na página 87.

EQUITRON SISTEMAS. *Manual do Sistema de exaustão Industrial Lamotriz.* Belém, Brasil, 2012. Citado na página 168.

ESC, M. *Serial Connection - Raspberry Pi to computer with UART Module*. 2015. Disponível em: <https://www.youtube.com/watch?v=owKEZTPDURs>. Acesso em: 17 mai. 2018. Citado na página 87.

FAIR. *Pytorch: An open source machine learning framework that accelerates the path from research prototyping to production deployment.* 2016. Disponível em: <http://pytorch.org/>. Acesso em: 26 jul. 2019. Citado na página 184.

FARHANGI, R.; BOROUSHAKI, M.; HOSSEINI, S. Load frequency control of interconnected power system using emotional learningbased intelligent controller. *International Journal of Electrical Power and Energy Systems.*, v. 36, p. 76–83, 2012. Citado na página 68.

FERRY B. ANDROOZENDAAL, B.; MCGAUGH, J. Role of norepinephrine in mediating stress hormone regulation of long-term memory storage: a critical involvement of the amygdala. *The Journal of J. Biolical Psychiatry.*, v. 46, p. 1140–1152, 1999. Citado na página 72.

FIORILLO, C. D.; YUN, S. R.; SONG, M. R. Diversity and homogeneity in responses of midbrain dopamine neurons. *The Journal of Neuroscience.*, v. 33, p. 4693–4709, 2013. Citado na página 65.

FOUNDATION, P. S. *Python.* 2020. Disponível em: <https://www.python.org/>. Acesso em: 11 ago. 2018. Citado na página 84.

FRANCKE, I. D.; FERREIRA, A. A. M.; GRASSI, R. Prazer e risco: As bases neuropsicologicas dos comportamentos motivados. *Reuniao Anual do Instituto de Neuropsicologia e Comportamento.*, p. 117, 2010. Citado 2 vezes nas páginas 66 e 67.

FU, K. Learning control systems-review and outlook. *Proceedings of The Space Congress.*, p. 9–23, 1970. Citado na página 27.

FUJIMOTO, S. S.; HOOF, H. van; MEGER, D. Addressing function approximation error in actor-critic methods. *ICML 2018 arXiv:1802.09477 [cs.AI]*, v. 3, 2018. Citado 2 vezes nas páginas 58 e 123.

FUSTER, J. M. *The prefrontal cortex: anatomy, physiology, and neuropsychology of the frontal lobe.* 4. ed. [S.l.]: Raven Press., 2006. Citado na página 74.

GEIBEL, P.; WYSOTZKI, F. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Arti*

cial Intelligence Research., v. 24, p. 81–108, 2005. Citado na página 112.

GERSHGORN, D. *New 'OpenAI' Artificial Intelligence Group Formed By Elon Musk, Peter Thiel, And More*. 2015. Disponível em: <https://www.popsci.com/new-openai-artificial-intelligence-group-formed-by-elon-musk-peter-thiel-and-more/>. Citado na página 90.

GOLDHIRSCH, I.; SULEM, P. L.; ORSZAG, S. A. Stability and lyapunov stability of dynamical systems: A differential approach and a numerical method. *Physica D: Nonlinear Phenomena*, v. 27, p. 311–337, 1987. Citado na página 109.

GONÇALVEZ, D. V. *Controle adaptativo de processo de nível utilizando aprendizado por reforço ator-crítico.* Dissertação — Universidade de Brasília., Brasília, Brasil, 2016. Citado na página 53.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *The Temporal Lobe and Limbic System.* 1. ed. [S.l.]: Oxford University Press., 1997. Citado na página 30.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning.* 1. ed. [S.l.]: MIT Press, 2016. Citado 2 vezes nas páginas 28 e 181.

GOODFELLOW, I. et al. Generative adversarial nets. *Advances in Neural Information Processing Systems.*, p. 2672–2680, 2014. Citado 2 vezes nas páginas 28 e 181.

GÖRGES D., . Relations between model predictive control and reinforcement learning. *IFAC-PapersOnLine 20th IFAC World.*, v. 50, p. 4920 – 4928, 2017. Citado na página 112.

GUELER, G. F. Modelling, design and analysis of an autopilot for submarine vehicles. *International Shipbuilding Progress*, v. 36, p. 51–85, 1989. Citado na página 120.

GUERRERO, A. D. M. *Inverted pendulum controlled via real-time techniques: plant development and modeling.* Dissertação (Trabalho de conclusão de curso) — Universidad técnica del norte, Ibarra, Equador, 2017. Citado na página 153.

GUYTON, A. C.; HALL, J. E. *Tratado de fisiologia médica.* 1. ed. [S.l.]: Elsevier Brasil., 2006. Citado 2 vezes nas páginas 71 e 76.

HA, H. et al. Correlation-based deep learning for multimedia semantic concept detection. *In International Conference on Web Information Systems Engineering.*, p. 473–487, 2015. Citado na página 28.

HAARNOJA, T. et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv*, p. eprint=1801.01290, 2018. Citado na página 61.

HAARNOJA, T. et al. Soft actor-critic algorithms and applications. *arXiv*, p. eprint=1812.05905, 2019. Citado na página 61.

HAFNER, D. et al. Dream to control: Learning behaviors by latent imagination. *arXiv:1912.01603v3 [cs.LG]*, 2020. Citado na página 65.

HAITH, A. M.; W., K. J. Model-based and model-free mechanisms of human motor learning. *Springer: Adv Exp Med Biol.*, v. 782, p. 1–21, 2013. Citado 2 vezes nas páginas 80 e 181.

HE, X.; ZHAO, K.; CHU, X. Automl: A survey of the state-of-the-art. *arXiv:1908.00709*, p. eprint 1908.00709, 2020. Citado na página 184.

HILL, R. *Prefrontal Cortex.* 2017. Disponível em: <https://www.thescienceofpsychotherapy.com/prefrontal-cortex/>. Acesso em: 25 jul. 2019. Citado na página 66.

HOOKER, C. I. et al. Amygdala response to facial expressions reflects emotional learning. *The Journal of Neuroscience.*, v. 26, p. 8915–8930, 2006. Citado 2 vezes nas páginas 72 e 73.

HOWARD, R. A. *Dynamic Programming and Markov Processes.* 1. ed. MA, USA: Princeton University Press, 1960. Citado 2 vezes nas páginas 25 e 26.

HSU, C. S. *Cell-to-Cell Mapping: A Method of Global Analysis for Nonlinear Systems.* 1. ed. [S.l.]: Springer, 1987. Citado na página 110.

JAFARI, E. et al. Designing an emotional intelligent controller for ipfc to improve the transient stability based on energy function. *J. of Elec. Eng. and Tech.*, p. 478–489, 2013. Citado na página 32.

JAFARI, M. et al. Brain emotional learning-based intelligent tracking control for unmanned aircraft systems with uncertain system dynamics and disturbance . *Proc. Int. Conf. Unmanned Aircraft Syst.*, p. 1470–1475, 2017. Citado na página 31.

JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, v. 22, p. 4–37, 2000. Citado na página 26.

JAMALI, M. R. et al. Real time emotional control for anti-swing and positioning control of simo overhead traveling crane. *International Journal of Innovative Computing, Information and Control.*, v. 4, p. 2333–2344, 2008. Citado na página 32.

JIANG Y., J. Z.-P. Computational adaptive optimal control for continuous-time linear systems with completely unknown dynamics. *Automatica 48.*, v. 10, p. 2699 – 2704, 2002. Citado na página 113.

JIN, M.; LAVAEI, J. Stability-certified reinforcement learning: A control-theoretic perspective. *ArXiv*, abs/1810.11505, 2018. Citado 2 vezes nas páginas 111 e 112.

KANDEL, E. R.; SCHWARTZ, J. H.; JESSELL, T. M. *Principles Of Neural Science.* 4. ed. [S.l.]: McGraw-Hill Medical., 2003. Citado 2 vezes nas páginas 72 e 73.

KELLY, J. P. *The neural basis of perception and movement. In Principles of neural science, pp. 283-295.* 3. ed. [S.l.]: Elsevier., 1991. Citado na página 75.

KHALIL, H. *Nonlinear Systems*. 1. ed. [S.l.]: Prentice-Hall, 1996. Citado na página 110.

KHORASHADIZADEH, S.; MAHDIAN, M. Voltage tracking control of dc-dc boost converter using brain emotional learning. *4th International Conference on Control, Instrumentation, and Automation*, p. 268–272, 2016. Citado na página 31.

KHORASHADIZADEH, S. et al. Robust model-free control of a class of uncertain nonlinear systems using belbic: stability analysis and experimental validation. *Journal of the Brazilian Society of Mechanical Sciences and Engineering*, v. 31, p. 84–93, 2019. Citado na página 109.

KIRCHHOFF, G. R. *Vorlesungen ueber Mathematische Physik, Mechanik.* 1. ed. Leipzig: BG Teubner Verlag was, 1876. Citado na página 120.

KLECKER, S.; HICHRI, B.; PLAPPER, P. Robust belbic-extension for trajectory tracking control. *J Mech Eng Autom*, v. 7, p. 84–93, 2017. Citado na página 109.

KLECKER, S.; PLAPPER, P. Belbic-sliding mode control of robotic manipulators with uncertainties and switching constraints. *Proceedings of the ASME, international mechanical engineering congress and exposition*, 2016. Citado na página 109.

KOLB, B.; WHISHAW, I. Q. *Fundamentals of human neuropsychology.* 6. ed. [S.l.]: Worth Publishers., 2009. Citado na página 74.

KRETCHMAR, R. M. et al. Robust reinforcement learning control with static and dynamic stability. *International Journal of Robust and Nonlinear Control*, v. 11, p. 1469–15001, 2001. Citado na página 54.

KUREMOTO, T. et al. A dynamic associative memory system by adopting amygdala model. *Artificial Life and Robotics.*, v. 13, p. 478–482, 2009. Citado na página 84.

KUREMOTO, T. et al. A functional model of limbic system of brain. *A Functional Model of Limbic System of Brain. In: Zhong N., Li K., Lu S., Chen L. (eds) Brain Informatics.*, v. 5819, p. 135–146, 2009. Citado na página 84.

LACHAUX, M. et al. Unsupervised translation of programming languages. *arXiv:2006.03511 [cs.CL]*, 2020. Citado na página 29.

LAKE, B. M. et al. Building machines that learn and think like people. *In press at Behavioral and Brain Sciences. arXiv:1604.00289 [cs.AI]*, 2016. Citado na página 55.

LANDAU, I. D. et al. *Adaptive Control: Algorithms, Analysis and Applications*. 2. ed. [S.l.]: Communications and Control Engineering. Springer-Verlag London, 2011. Citado 2 vezes nas páginas 111 e 112.

LAUTIN, A. L. *The Limbic Brain.* 1. ed. [S.l.]: Springer., 2001. Citado na página 30.

LEDOUX, J.; FELLOUS, J. M. *Emotion and computational neuroscience. In: Handbook of brain theory and neural networks.* [S.l.]: MIT Press., 1995. Citado 5 vezes nas páginas 70, 76, 77, 183 e 184.

LENDARIS, G. G. A retrospective on adaptive dynamic programming for control. *Proceedings of International Joint Conference on Neural Networks.*, p. 1750–1757, 2009. Citado na página 26.

LEVINE, S. et al. Endtoend training of deep visuomotor policies. *JMLR.*, v. 17, p. 1–40, 2016. Citado na página 29.

LEVINE, S. et al. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *ISER.*, 2016. Citado na página 29.

LEWIS, F. L.; VRABIE, D.; VAMVOUDAKIS, K. G. Reinforcement learning and feedback control: Using natural decision methods to design optimal adaptive controllers. *IEEE Control Systems.*, v. 32, p. 76–105, 2012. Citado 2 vezes nas páginas 112 e 113.

LEWONTIN, M. *Open AI: Effort to democratize artificial intelligence research?* 2015. Disponível em: <https://www.csmonitor.com/Technology/2015/1214/Open-AI-Effort-to-democratize-artificial-intelligence-research>. Citado na página 90.

LI, K.; MALIK, J. Learning to optimize. *ICLR*, 2017. Citado na página 55.

LI, Y. Deep reinforcement learning: An overview. *ArXiv*, ArXiv:1701.07274, 2017. Citado 2 vezes nas páginas 111 e 114.

LILLICRAP, T. P. et al. Continuous control with deep reinforcement learning. *ICLR*, 2016. Citado 8 vezes nas páginas 48, 57, 58, 65, 111, 112, 119 e 183.

LJUNG, L. *System Identification: Theory for the User.* 2. ed. [S.l.]: Prentice Hall., 1999. Citado 2 vezes nas páginas 113 e 114.

LOTFI, E.; KHAZAEI, O.; KHAZAEI, F. Competitive brain emotional learning. *Neural Process. Lett*, v. 47, p. 745–764, 2017. Citado 2 vezes nas páginas 30 e 119.

LOTFI, E.; KHOSRAVI, A.; NAHAVANDI, S. Facial emotion recognition using emotional neural network and hybrid of fuzzy c-means and genetic algorithm. *IEEE Int. Conf. Fuzzy Syst.*, p. 16, 2017. Citado na página 30.

LOTFI, E.; REZAEE, A. A. Generalized belbic. *The Natural Computing Applications.*, v. 14, p. 4367–4383, 2018. Citado 7 vezes nas páginas 32, 99, 109, 119, 142, 146 e 181.

LUCAS, C.; RASHIDI, F.; ABDI, J. Transient stability improvement in power systems via firing angle control of tcsc using context based emotional controller. *IEEE. Automation Congress.*, v. 16, p. 37–42, 2004. Citado 2 vezes nas páginas 32 e 68.

LUCAS, C.; SHAHMIRZADI, D.; SHEIKHOLESLAMI, N. Introducing belbic: brain emotional learning based intelligent controller . *Intelligent Automation and Soft Computing, TF.*, v. 10, p. 11–21, 2004. Citado 10 vezes nas páginas 32, 33, 68, 81, 83, 84, 99, 111, 127 e 181.

LUNDAGARD, K.; BALKENIUS, C. A computational model of emotional learning in the amygdala. *in From Animals to Animats 6: Proceedings of the Sixth International Conference on Simulation of Adaptive Behavior.*, 2000. Citado na página 30.

MACLEAN, P. Psychosomatic disease and the visceral brain; recent developments bearing on the papez theory of emotion. *Psychosom Med.*, v. 11, p. 338–53, 1949. Citado na página 71.

MACLEAN, P. Some psychiatric implications of physiological studies on frontotemporal portion of limbic system (visceral brain). *Electroencephalography and Clinical Neurophysiology.*, v. 4, p. 407–418, 1952. Citado na página 71.

MANDZUKA, S. Mathematical model of a submarine dynamics at the periscope depth. *Brodogradnja : casopis brodogradnje i brodogra djevne industrije*, v. 46, p. 129–139, 1998. Citado na página 120.

MARKADEH, G. R. et al. Speed and flux control of induction motors using emotional intelligent controller. *IEEE Transactions on Industry Applications.*, v. 47, p. 1126–1135, 2011. Citado 5 vezes nas páginas 33, 68, 81, 99 e 181.

MATHWORKS. *Simulation and Model-Based Design*. 2019. Disponível em: <https://www.mathworks.com/products/simulink.html>. Citado 2 vezes nas páginas 84 e 91.

MATHWORKS. *Reinforcement Learning Toolbox.* 2020. Disponível em: <https://www.mathworks.com/products/reinforcement-learning.html>. Acesso em: 15 ago. 2018. Citado 2 vezes nas páginas 52 e 53.

MATIISEN, T. *Demystifying deep reinforcement learning*. 2017. Disponível em: <http://neuro.cs.ut.ee/demystifying-deep-reinforcement-learning>. Acesso em: 08 jun. 2019. Citado 2 vezes nas páginas 56 e 65.

MEDICALSCHOOL. *Amígdala.* 2018. Disponível em: <https://medicalschool.tumblr.com>. Acesso em: 11 jul. 2019. Citado na página 75.

MEHRABIAN, A. R.; LUCAS, C.; ROSHANIAN, J. Aerospace launch vehicle control: an intelligent adaptive approach. *Aerospace Science and Technology.*, v. 2, p. 149–155, 2006. Citado na página 68.

MENDEL, J. M. A survey of learning control systems. *ISA Transactions.*, v. 5, p. 297–303, 1966. Citado 2 vezes nas páginas 26 e 27.

MENDEL, J. M.; MACLAREN, R. W. Reinforcement learning control and pattern recognition systems. *Adaptive, Learning, and Pattern Recognition Systems:Theory and Applications.*, p. 287–318, 1970. Citado na página 27.

MINSKY, M. L. Steps toward artificial intelligence. *Proceedings of the Institute of Radio Engineers, Reprinted in Computers and Thought* ., v. 1, p. 406–450, 1963. Citado 2 vezes nas páginas 26 e 27.

MIRHAJIANMOGHADAM, H.; AKBARZADEH, M. R.; LOT, E. A harmonic emotional neural network for non-linear system identication. *Proc.Iranian Conf. Electr. Eng.*, p. 1260–1265, 2016. Citado na página 30.

MNIH, V. et al. Asynchronous methods for deep reinforcement learning. *arXiv*, p. eprint=1602.01783, 2016. Citado na página 63.

MNIH, V. et al. Playing atari with deep reinforcement learning. *NIPS Deep Learning Workshop. arXiv:1312.5602 [cs.LG]*, 2013. Citado 2 vezes nas páginas 56 e 112.

MNIH, V. et al. Human-level control through deep reinforcement learning. *Nature*, v. 518, p. 529–533, 2015. Citado 3 vezes nas páginas 55, 65 e 119.

MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of Machine Learning.* 1. ed. NY, USA: MIT Press, 2012. Citado na página 38.

MOON, G.; L., L. K. I2l-meshnet: Image-to-lixel prediction network for accurate 3d human pose and mesh estimation from a single rgb image. *arXiv:2008.03713v2 [cs.CV]*, 2020. Citado na página 29.

MORéN, J. Emotion and learning: A computational model of the amygdala. *Lund University Cognitive Science*, v. 93, p. 16, 2002. Citado 6 vezes nas páginas 32, 68, 78, 79, 183 e 184.

MORéN, J.; BALKENIUS, C. A. Emotion and learning: A computational model of the amygdala. *From Animals to Animats: The sixth international conference on the simulation of adaptive behavior*, v. 6, p. 115–124, 2000. Citado 3 vezes nas páginas 32, 68 e 78.

NAJAFABADI, M. M. et al. Deep learning applications and challenges in big data analytics. *Journal of Big Data 2.*, v. 1, p. 1–21, 2015. Citado 2 vezes nas páginas 29 e 129.

NARDELLA, D. *Step7 Open Source Ethernet Communication Suite.* 2012. Disponível em: <http://snap7.sourceforge.net/>. Acesso em: 16 out. 2019. Citado na página 97.

NIELSEN, M. A. *Neural Networks and Deep Learning.* 1. ed. [S.l.]: Determination Press, 2015. Citado na página 27.

NVIDIA. *Develop, Optimize and Deploy GPU-accelerated Apps*. 2019. Disponível em: <https://developer.nvidia.com/cuda-toolkit>. Citado na página 88.

NVIDIA. *GPU Accelerated Computing*. 2019. Disponível em: <https://www.nvidia.com/en-us/about-nvidia/ai-computing/>. Citado na página 88.

NVIDIA. *NVIDIA cuDNN*. 2019. Disponível em: <https://developer.nvidia.com/cudnn>. Citado na página 88.

OGATA, K. *Engenharia de Controle Moderno.* 4. ed. Rio de Janeiro, Brasil: Prentice-Hall, 2003. Citado 8 vezes nas páginas 25, 95, 108, 111, 114, 119, 152 e 182.

OHMAN, A.; MINEKA, S. Fears, phobias, and preparedness: Toward an evolved module of fear and fear learning. *Psychological Review.*, v. 108, p. 483–522, 2001. Citado 3 vezes nas páginas 75, 183 e 184.

O'KEEFE, J.; NADEL, L. *The hippocampus as a cognitive map.* 1. ed. Oxford, UK: Clarendon Press., 1978. Citado na página 76.

OPENAI. *Welcome to Spinning Up in Deep RL.* 2020. Disponível em: <https://spinningup.openai.com/>. Acesso em: 18 set. 2018. Citado 3 vezes nas páginas 60, 61 e 62.

PADOA-SCHIOPPA, C.; ASSAD, J. A. Neurons in the orbitofrontal cortex encode economic value . *Nature.*, v. 441, p. 223–226, 2006. Citado na página 30.

PANKSEPP, J. *Hypothalamic integration of behavior: Rewards, punishments, and related psychobiological processes.* NY, USA: Marcel Dekker., 1981. Citado na página 70.

PAPEZ, J. A proposed mechanism of emotion. *Archives of Neurological Psychiatry.*, v. 38, p. 4725–743, 1937. Citado 2 vezes nas páginas 69 e 70.

PARSAPOOR, M.; LUCAS, C.; SETAYESHI, S. Reinforcement recurrent fuzzy rule based system based on brain emotional learning structure to predict the complexity dynamic system. *Third IEEE International Conference on Digital Information Management.*, p. 25–32, 2008. Citado na página 84.

PAVLOV, I. *Conditional reflexes: An investigation of the physiological activity of the cerebral cortex.* 2. ed. Oxford, UK: Oxford University Press, 1927. Citado 2 vezes nas páginas 26 e 28.

PERKINS, T. J.; BARTO, A. G. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research.*, v. 3, p. 803–832, 2002. Citado na página 109.

PEZZULO, G. et al. Internally generated sequences in learning and executing goal-directed behavior . *Trends in Cognitive Science.*, v. 18, p. 647–657, 2014. Citado na página 31.

POUYANFAR, S. et al. A survey on deep learning: Algorithms, techniques, and applications. article 92. *ACM Comput. Surv.*, v. 5, 2018. Citado 2 vezes nas páginas 29 e 119.

PRECUP, D.; SUTTON, R. S.; DASGUPTA, S. Y. Off-policy temporal-difference learning with function approximation. *Proceedings of the 18th International Conference on Machine Learning.*, p. 417–424, 2001. Citado na página 27.

PURVES, D. et al. *Neuroscience.* 5. ed. Oxford, England: Sinauer Associates, Inc., 2011. Citado na página 72.

PUTERMAN, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming.* 2. ed. NJ, USA: John Wiley and Sons, Inc., 1994. Citado na página 40.

RAFFIN, A. *RL Baselines Zoo*. 2018. Disponível em: <https://github.com/araffin/rl-baselines-zoo>. Acesso em: 26 fev. 2020. Citado na página 155.

RAHMAN, M. A. et al. Implementation of emotional controller for interior permanent-magnet synchronous motor drive . *IEEE Transactions on Industry Applications.*, v. 44, p. 1466–1476, 2008. Citado 3 vezes nas páginas 31, 32 e 68.

RANGEL, A.; CAMERER, C.; MONTAGUE, P. R. A framework for studying the neurobiology of value-based decision making. *Nature Reviews Neuroscience.*, v. 9, p. 545–556, 2008. Citado na página 30.

RANGEL, A.; HARE, T. Neural computations associated with goal-directed choice. *Current Opinion in Neurobiology .*, v. 20, p. 262–270, 2010. Citado na página 30.

RASPBERRY, F. *Faqs Raspberry PI*. 2019. Disponível em: <https://www.raspberrypi.org/faqs#softwaresLanguages>. Citado 3 vezes nas páginas 87, 168 e 182.

ROBERTS, A. Primate orbitofrontal cortex and adaptive behavior. *Trends in cognitive sciences.*, v. 10, p. 83–90, 2006. Citado na página 73.

ROBOCORE. *Comparação Entre Protocolos de Comunicação Serial*. 2016. Disponível em: <https://www.robocore.net/tutoriais/comparacao-entre-protocolos-de-comunicacao-serial.html>. Citado na página 92.

ROLLS, E. T. A theory of emotion and consciousness, and its application to understanding the neural basis of emotion. *Cognition and Emotion.*, v. 4, p. 161–190, 1990. Citado 2 vezes nas páginas 76 e 77.

ROUHANI, H. et al. Brain emotional learning based intelligent controller applied to neurofuzzy model of micro-heat exchanger. *Expert Systems Applications.*, v. 3, p. 911–918, 2007. Citado na página 68.

RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence: A Modern Approach.* 3. ed. [S.l.]: Prentice Hall ., 2010. Citado 2 vezes nas páginas 37 e 38.

SADEGHI, M. A.; DARYABEIGI, E. Real-time implementation of brain emotional learning developed for digital signal processor-based interior permanent magnet synchronous motor drive systems. *Journal of power electronics.*, v. 14, p. 74–81, 2014. Citado 3 vezes nas páginas 31, 99 e 181.

SADEGHIEH, A.; ROSHANIAN, J.; NAJA, F. Implementation of an intelligent adaptive controller for an electrohydraulic servo system based on a brain mechanism of emotional learning. *International Journal of Advanced Robotic Systems*, v. 9, p. 1–12, 2012. Citado na página 31.

SARANT, H. G.; NETSKY, M. G. *Evolution of the nervous system.* 1. ed. Oxford, England: Oxford U. Press., 1974. Citado 2 vezes nas páginas 71 e 72.

SCHACHTER, S. Some extraordinary facts about obese humans and rats. *American Psychologist, American Psychological Association.*, v. 26, p. 129, 1971. Citado na página 76.

SCHOKNECHT, R.; MERKE, A. Convergent combinations of linear function approximation. *Advances in Neural Information Processing Systems.*, p. 1579–1586, 2003. Citado na página 27.

SCHRITTWIESER, J. et al. Mastering atari, go, chess and shogi by planning with a learned model. *arXiv:1911.08265v2 [cs.LG]*, 2020. Citado na página 65.

SCHULMAN, J. et al. Trust region policy optimization. *arXiv:1502.05477 [cs.LG].*, 2015. Citado 2 vezes nas páginas 59 e 123.

SCHULMAN, J. et al. High-dimensional continuous control using generalized advantage estimation . *arXiv*, p. eprint=1506.02438, 2018. Citado na página 59.

SCHULMAN, J. et al. Proximal policy optimization algorithms. *arXiv:1707.06347 [cs.LG].*, 2017. Citado 2 vezes nas páginas 61 e 123.

SHAHMIRZADI, D. *Computational modeling of the brain limbic system and its application in control engineering.* Dissertação (Tese de doutorado) — Texas A M University, USA, 2005. Citado 7 vezes nas páginas 81, 118, 119, 120, 126, 132 e 183.

SHARBAFI, M. A.; LUCAS, C.; DANESHVAR, R. Motion control of omni-directional three-wheel robots by brain-emotional-learningbased intelligent controller. *EEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews).*, v. 40, p. 630–638, 2010. Citado 5 vezes nas páginas 33, 68, 81, 99 e 181.

SHIMAMURA, A. P. *Memory and frontal lobe function.* 1. ed. [S.l.]: MIT Press., 1995. Citado na página 74.

SIEMENS. *Já pensou em Machine Learning e Inteligência Artifical em máquina? Com SIMATIC isso é possível.* 2019. Disponível em: <https://new.siemens.com/br/pt/produtos/automacao/webinars.html>. Acesso em: 01 agos. 2019. Citado na página 39.

SILVA, E. A. *Técnicas de estimação de parâmetros de módulos fotovoltaicos*. Dissertação (Dissertação de Mestrado) — Progama de Pós Graduação em Engenharia Elétrica, Universidade Federal de Pernambuco, Recife, Brasil, 2015. Citado 2 vezes nas páginas 49 e 174.

SILVA, J. L. *Controle eficiente com ferramentas de inteligência artificial em um sistema de exaustão.* Dissertação (Dissertação de mestrado) — Universidade Federal de Pernambuco, Recife, Brasil, 2017. Citado na página 170.

SILVA, J. L. et al. A plc-based fuzzy logic control with metaheuristic tuning. *Studies in Informatics and Control, ISSN 1220-1766.*, v. 28(3), p. 265–278, 2019. Citado 2 vezes nas páginas 166 e 174.

SILVER, D. et al. Mastering the game of go with deep neural networks and tree search. *Nature*, v. 529, p. 484–489, 2016. Citado na página 55.

SUN, R.; BOOKMAN, L. *Computational Architectures Integrating Neural and Symbolic Processes.* 3 p. 336. ed. [S.l.]: Springer US., 1994. Citado na página 66.

SUTTON, R. S. *Temporal credit assignment in reinforcement learning*. Dissertação (tese de doutorado) — University of Massachusetts Amherst, MA , EUA, 1984. Citado 3 vezes nas páginas 26, 111 e 165.

SUTTON, R. S. Learning to predict by the method of temporal differences. *Machine Learning.*, v. 3, p. 9–44, 1988. Citado 2 vezes nas páginas 26 e 27.

SUTTON, R. S. Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in Neural Information Processing Systems: Proceedings of the 1995 Conference.*, p. 1038–1044, 1996. Citado na página 27.

SUTTON, R. S.; BARTO, A. G. *Reinforcement Learning, An Introduction.* 2. ed. London, England: MIT Press, 2018. Citado 12 vezes nas páginas 25, 26, 28, 29, 40, 41, 42, 43, 44, 46, 47 e 181.

SUTTON, R. S.; BARTO, A. G.; WILLIAMS, R. J. Reinforcement learning is direct adaptive optimal control. *IEEE Control Systems.*, v. 12, p. 19–22, 1992. Citado na página 28.

TEIXEIRA, H. T. *Aprendizado por reforço e programação dinâmica aproximada com máquinas Kernel para controle de sistemas não lineares.* Dissertação (Tese de doutorado) — Universidade Estadual de Campinas, Campinas , Brasil, 2016. Citado 2 vezes nas páginas 45 e 47.

TENSORFLOW. *Introduction to TensorFlow*. 2019. Disponível em: <https://www.tensorflow.org/learn>. Citado na página 86.

TESAURO, G. Practical issues in temporal difference learning. *Machine Learning.*, v. 8, p. 257–277, 1992. Citado na página 27.

THORNDIKE, E. L. *Animal Intelligence: An Experimental Study of the Associative Processes in Animals, sér. Psychological Rewiew, Monograph Supplements 4* . 2. ed. NY, USA: The MacMillan Company, 1898. Citado na página 26.

TOLEDO, D. *Cortex motor e sensorial.* 2009. Disponível em: <http://sistemaendocrinounisul.blogspot.com/>. Acesso em: 14 jul. 2019. Citado na página 75.

TOLLIVER, J. V. . *Studies on Submarine Control for Periscope Depth Operations*. Dissertação (Master's Thesis,) — Naval Postgraduate School, Monterey., 1996. Citado na página 120.

TSINIAS, J.; KALOUPTSIDIS, N.; BACCIOTTI, A. Lyapunov functions and stability of dynamical polysystems. mathematical systems theory. *Physica D: Nonlinear Phenomena*, v. 19, p. 333–354, 1986. Citado 2 vezes nas páginas 109 e 110.

TSITSIKLIS, J. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control.*, v. 42, p. 674–690, 1997. Citado na página 27.

TSITSIKLIS, J. N.; ROY, B. V. Feature-based methods for large scale dynamic programming. *Machine Learning.*, v. 22, p. 59–94, 1996. Citado na página 27.

TU, J. *Continuous Reinforcement Learning for Feedback Control Systems.* Dissertação (M.S. Thesis) — Department of Computer Science, Colorado State University., Fort Collins, USA, 2001. Citado na página 54.

TURING, A. M. Computing machinery and intelligence. *Mind*, p. 433–460, 1950. Citado na página 27.

UHLENBECK, G. E.; ORNSTEIN, L. S. On the theory of brownian motion. *Phys. Rev.*, v. 36, p. 823–841, 1930. Citado na página 58.

UNICONTROL. *Curso PLC Siemens Módulo básico usanto o software STEP 7.* 1. ed. [S.l.: s.n.], 2003. Citado na página 169.

VALENTIN, V. V.; DICKINSON, A.; O'DOHERTY, J. P. Determining the neural substrates of goal-directed learning in the human brain. *The Journal of Neuroscience.*, v. 27, p. 4019–4026, 2007. Citado na página 30.

VAPNIK, V. N. *The Nature of Statistical Learning Theory.* 2. ed. NY, USA: Springer Verlag, 2000. Citado na página 38.

WALTZ, M. D.; FU, K. A heuristic approach to reinforcement learning control. *IEEE Transactions on Automatic Control.*, v. 10, p. 390–398, 1965. Citado 2 vezes nas páginas 26 e 27.

WANG, J. et al. Arma model identification using particle swarm optimization algorithm. *Proceedings of the International Conference on Computer Science and Information Technology*, p. 223–227, 2008. Citado na página 25.

WANG, J. X. et al. Learning to reinforcement learn. *CogSci.*, 2017. Citado na página 29.

WANG, Z. et al. Sample efficient actor-critic with experience replay. *arXiv*, p. eprint=1611.01224, 2017. Citado na página 64.

WATKINS, C. J.; DAYAN, P. Technical note q-learning. *Machine Learning.*, v. 8, p. 279–292, 1992. Citado na página 49.

WATKINS, C. J. C. H. *Learning from delayed rewards.* Dissertação (tese de doutorado) — King's College, Cambridge, UK, 1989. Citado 2 vezes nas páginas 26 e 48.

WERBOS, P. J. Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *IADS Newsletter IEEE Transactions on Systems, Man and Cybernetics.*, v. 17, p. 7–20, 1987. Citado na página 26.

WILLIAMS, J. K. Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *Artificial intelligence methods in the environmental sciences* ., v. 1, p. 297–327, 2009. Citado na página 26.

WILLIAMS, J. K. Reinforcement learning of optimal control. *Artificial intelligence methods in the environmental sciences.*, p. 297–327, 2009. Citado 2 vezes nas páginas 27 e 28.

WU, Y. et al. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *arXiv*, p. eprint=1708.05144, 2017. Citado na página 65.

WU, Y. et al. Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. *arXiv:1708.05144 [cs.LG].*, 2017. Citado na página 123.

XU, G.; GAO, J. Weak signal detection based on a new matching pursuit method. *Proceedings of the 7th International Conference on Machine Learning and Cybernetics*, v. 17, p. 406–413, 2008. Citado na página 27.

YAN, Y. et al. Efficient imbalanced multimedia concept retrieval by deep learning on spark clusters. *International Journal of Multimedia Data Engineering and Management.*, v. 8, p. 1–20, 2017. Citado na página 29.

YAN, Y. et al. Deep learning for imbalanced multimedia data classification. *IEEE International Symposium on Multimedia.*, p. 483–488, 2015. Citado na página 29.

YI, H. A sliding mode control using brain limbic system control strategy for a robotic manipulator . *Inter. Journal of Advanced Robotic Systems*, v. 12, p. 1–9, 2015. Citado na página 31.

YOON, C. *Understanding Actor Critic Methods and A2C.* 2019. Disponível em: <https://towardsdatascience.com/understanding-actor-critic-methods-931b97b6df3f>. Acesso em: 18 dez. 2019. Citado na página 63.

ZHOU, D. et al. Fuzzy brain emotional learning control system design for nonlinear systems . *Int. J. Fuzzy Syst.*, v. 17, p. 117–128, 2015. Citado na página 32.

ZHOU, D. et al. Integration of fuzzycmac and belc networks for uncertain nonlinear system control. *Proc. IEEE Int. Conf. Fuzzy Syst. (FUZZ-IEEE)*, p. 16, 2017. Citado na página 32.

ZOPH, B.; LE, Q. V. Neural architecture search with reinforcement learning. *ICLR*, 2017. Citado na página 55.

APÊNDICE A – CÓDIGOS FONTES

```
#DBELBIC


class DBEL:
    def __init__(self):

        self.v = 0.0
        self.w = 0.0
        self.dv = 0.0
        self.dw = 0.0
        self.alpha0 = 0.0001
        self.beta0 = 0.0001
        self.rd = 0.05
        self.ts = 0.0

    def DBelbic(self, S, R, limite):

        _limit_out = limite

        A = self.v*S  #S - A partir das DNNs
        O = self.w*S  #R - A partir das DNNs

        if R != 0:
            Ro = max(0,(A-R))-O
        else:
            Ro = max(0,(A-O))
        self.ts = self.ts + 0.01
        self.alpha = (1/ (1 + self.rd*self.ts))*self.alpha0
        self.beta = (1/ (1 + self.rd*self.ts))*self.beta0

        dv = self.alpha*S*(max(0,(R - A)))
        dw = self.beta*S*Ro

        self.v = self.v + dv
        self.w = self.w + dw

        MO = A - O

        if MO  > _limit_out:
            MO  = _limit_out
        elif MO  < - _limit_out:
            MO  = - _limit_out

        return MO, self.v, self.w
```

```
#Exemplo de Treinamento e Simulação de um agente TD3 no ambiente do pêndulo invertido


import gym
import numpy as np
from BEL.BEL_controller import BEL
from stable_baselines import TD3
from stable_baselines.td3.policies import MlpPolicy
from stable_baselines.ddpg.noise import NormalActionNoise, OrnsteinUhlenbeckActionNoise

DBEL = DBEL().DBelbic

env = gym.make('CartPoleContinuous-v0') #Construção do ambiente dinâmico



n_actions = 1
action_noise = NormalActionNoise(mean=np.zeros(n_actions), sigma=0.1 * np.ones(n_actions))

model = TD3(MlpPolicy, env, action_noise=action_noise, verbose=1, tensorboard_log="./TD3/")

model.learn(total_timesteps=2000000, log_interval=1000) #Realização do treinamento


#Simulação do agente treinado

obs = env.reset()


for i in range(10000):
    actions, _states = model.predict(obs)
    S = actions[0]
    R = actions[1]
    u,v,w = DBEL(S,R,50)
    obs, rewards, dones, info = env.step(u)

env.close()
```

```python
#Comunicação serial para o ambiente dinâmico em Simulink
import gym
from gym import spaces
from gym.utils import seeding
import numpy as np
from os import path
import serial
import struct
import time

class simulink(gym.Env):
    metadata = {
        'render.modes' : ['human', 'rgb_array'],
        'video.frames_per_second' : 30
    }
    def __init__(self):
        self.acao = 100 #Valor limite da ação no ambiente
        maximo = np.array([100,100, 100]) #erro, int_erro, der_erro
        self.action_space = spaces.Box(low=0, high=self.acao, shape=(1,), dtype=np.float32)
        self.observation_space = spaces.Box(-maximo,maximo, dtype=np.float32)
        self.ser = serial.Serial('COM20', baudrate =256000) #Abrir canal de comunicação serial
        self.ser.reset_input_buffer()
        self.ser.reset_output_buffer()
        self.viewer = None
        self.seed()

    def seed(self, seed=None):
        self.np_random, seed = seeding.np_random(seed)
        return [seed]

    def step(self,u):
        u # [K1,K2,K3,K4,K5,K6,K7]
        #Aplicar o sinal de controle (Ganhos) no Simulink via serial
        control = struct.pack('%sd' % len(u), *u)# array serial
        self.ser.write(control)
        time.sleep(0.1)
        data = self.ser.read(72) #Resposta do Simulink via serial
        datas = struct.unpack('<ddddddddd',data)
        self.state = np.array([datas[0],datas[1],datas[2]]) #erro, int_erro, der_erro
        reward, done = datas[7] , datas[8]
        return self.state, reward, done, {}

    def reset(self):
        #Aplicar o comando para resetar o environment no Simulink
        u = [0,0,0,0,0,0,0,1] #Reset ambiente Simulink
        control = struct.pack('%sd' % len(u), *u)# array serial
        self.ser.write(control)
        time.sleep(0.1)
        data = self.ser.read(72) #Resposta do Simulink via serial
        datas = struct.unpack('<ddddddddd',data)
        self.state = np.array([datas[0],datas[1],datas[2]]) #erro, int_erro, der_erro
        reward, done = datas[7] , datas[8]
        return self.state

    def close(self):
        if self.viewer:
            self.viewer.close()
            self.ser.close()
            self.viewer = None
```

```
#DBELBIC via Snap7

import math
import time
import serial
import struct
import numpy as np
import warnings
import snap7.client
from snap7.snap7types import *
from snap7.util import *
from DBELBIC import DBEL
from stable_baselines import TD3
from simple_pid import PID #https://github.com/m-lundberg/simple-pid
DBEL = DBEL()
global x
global vv
global ww
x = []
vv = []
ww = []
model = TD3.load("td3_version_3") #Agente TD3

last_time = 0
if __name__ == "__main__":

    while True:
        client= snap7.client.Client() #Comunicação Snap7
        client.connect('150.161.52.3',0,2) #Abrir canal de comunicação com o CLP

        # Ler condição para habilitar o controle a partir do supervisório
        data = client.db_read(12,0,12)
        control = get_bool(data,2,0)
        print('Aguarde o comando via Supervisório')
        while control == True: #Controle ativado via supervisório
            data1 = client.db_read(12,0,12) #Leitura da DB12
            data2 = client.db_read(1,0,300) #Leitura da DB1
            data3 = client.db_read(2,0,300) #Leitura da DB2

            setpoint = get_real(data1,4)
            erro = get_real(data1,8)
            vazao = get_real(data2,238)
            control = get_bool(data1,2,0)
            setfreq = get_real(data3,202)

            #Obter as parcelas proporcional, integral e derivada do sinal do erro
            pid_obs = PID(1, 1, 1)
            pid_obs.setpoint = setpoint
            pid_obs(vazao)
            erro, int_erro, der_erro = pid_obs.components
            obs = np.array([erro, int_erro, der_erro])

            K1,K2,K3 =  model.predict(obs) #Agente de DRL

            pid = PID(1, K2)
            pid.setpoint = setpoint
            pid(vazao)
            P,I,D = pid.components #Buscar a componente do integrador (I)


            R = K1*erro + I #Estímulo Emocional
```

```
S = K3 #Estímulo Sensorial
u,v,w = DBEL.DBelbic(S,R,100) #DBELBIC

#Limitar o sinal de controle
if u > 60:
    u = 60
if u < 0:
    u = 0

real = struct.pack('>f',u)
client.db_write(2,202,real) #Escrever o sinal de controle no CLP
last_time = time.time()
print('Erro:',erro, 'Vazão:',vazao)
print('SETPOINT:',setpoint,'freq:',u)
x.append(u)
vv.append(v)
ww.append(w)
```

APÊNDICE B – PARÂMETROS E HIPERPARÂMETROS

Parâmetros e Hiperparâmetros	**Ambientes Dinâmicos**			
	Submarino	**Braço Robótico**	**Pêndulo Invertido**	**Exaustor Industrial**
Arquitetura	AC	AC	AC	AC
Política	MlpPolicy	MlpLnLstmPolicy	RL-ZOO*	MlpPolicy
Nº de Camadas (Comum|Ator|Crítico)	40|50|50	60|80|80	RL-ZOO*	40|50|50
Nº de Unidades (Comum|Ator|Crítico)	32|216|64	64|256|128	RL-ZOO*	32|216|64
Learning Rate	0.001	0.00025	0.00025	0.001
Discount Factor	0.99	0.99	0.99	0.99
Minibatch Size	32	32	32	32
Otimização	Adam	Adam	Adam	Adam
Replay Buffer	6000	8000	8000	6000
Agente	TD3	PPO	Biblioteca *OpenAI*	PPO-LSTM

RL-ZOO*: https://stable-baselines.readthedocs.io/en/master/guide/rl_zoo.html

Algoritmos originais: https://github.com/openai/baselines/tree/master/baselines

APÊNDICE C – RASPBERRY PI - PC

Uma etapa importante no desenvolvimento e comissionamento do controlador proposto é a comunicação física entre os diferentes *hardwares*, os quais contêm o próprio controlador e o respectivo sistema dinâmico a ser controlado. Essa importância se dá pelo fato de que o agente de controle e o seu respectivo ambiente dinâmico necessitam estar em sincronia, referindo-se à transmissão das informações entre ambos, ou seja, ações do controlador e as observações e recompensas advindas do ambiente. A sincronia adequada entre os *hardwares* proporciona ações de controle efetivas, pois caso contrário, dificilmente os parâmetros do controlador serão satisfatórios para o seu funcionamento do sistema de controle.

Uma das principais etapas de comissionamento do controlador visa realizar a implementação de um agente controlador DRL via *hardware Raspeberry Pi* e, utilizá-lo para o controle de sistemas dinâmicos, em sua maioria, restritos ao ambiente de simulação computacional. Nesse caso, a comunicação entre o *hardware Raspeberry Pi* e o ambiente computacional ocorre por meio do protocolo de comunicação *serial*[1] *UART*[2]. Nesse sentido, vale salientar que a vantagem deste tipo de comunicação se encontra na simplicidade desse protocolo, onde o envio e recebimento de toda a informação acontece de maneira sequencial *full-duplex*[3], permitindo assim utilizar uma quantidade menor de fios.

O protocolo de comunicação UART é do tipo assíncrono, ou seja, não possui um sinal de *clock* associado, o que pode levar a erros na comunicação (ROBOCORE, 2016). Por este motivo, um parâmetro importante denominado *baud rate* precisa ser adicionado.

De forma simples, o *baud rate* é responsável por especificar a taxa de velocidade do envio e recebimento das informações entre os dispositivos comunicantes. Nesse caso, tais dispositivos, ligados na mesma linha, devem estar com o mesmo *baud rate*, pois do contrário dificilmente ocorrerá a transmissão adequada dos dados (ROBOCORE, 2016). No âmbito deste trabalho, após simulações e testes de comunicação, o *baud rate* escolhido foi de 9600 bps[4], o que equivale a uma taxa de transmissão dos dados de aproximadamente um bit a cada 0,0001s.

No que diz respeito à ligação física deste tipo de comunicação, o pino de transmissão (TX) deve ser conectado ao pino receptor (RX) do outro dispositivo, ou seja, um transmissor enviando para outro receptor, e vice-versa. No caso dos outros terminais (VCC, GND), ambos devem ser conectados com os respectivos semelhantes.

A Figura 107 apresenta um esquema de ligação e funcionamento da comunicação UART utilizada neste trabalho.

---

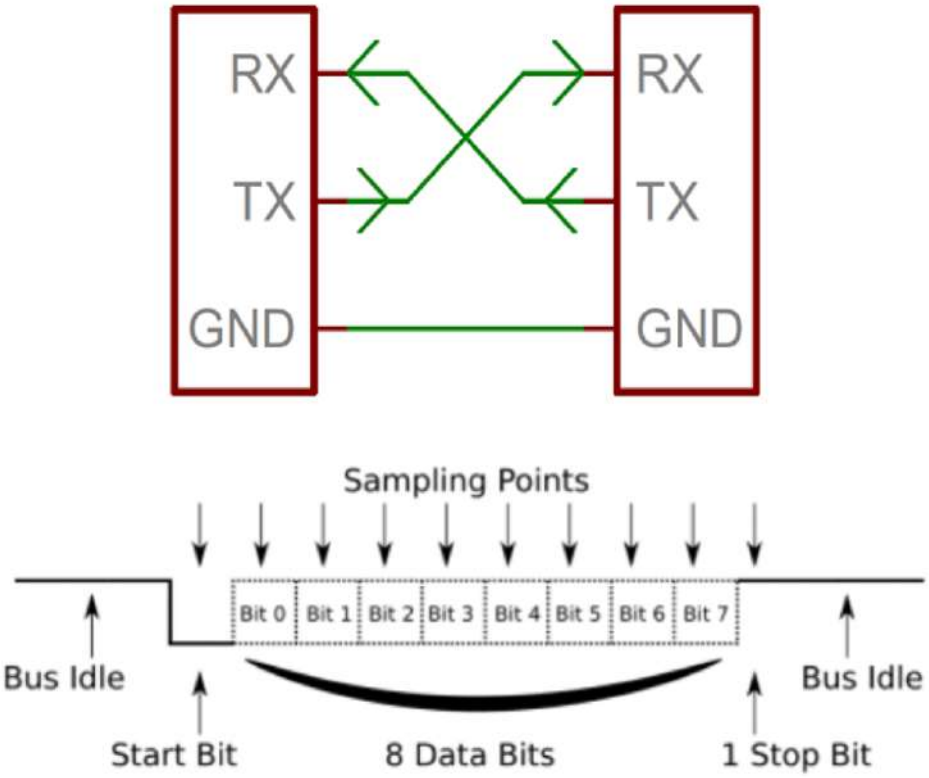
[1] Processo de comunicação sequencial, onde envia-se os dados por meio de um bit de cada vez, sequencialmente, num canal de comunicação ou barramento.

[2] Abreviação do inglês Universal Asynchrounous Receiver/Transmiter ou Receptor/Transmissor Universal Assíncrono.

[3] Indica que o dispositivo pode transmitir e receber dados ao mesmo tempo.

[4] Na área da comunicação esta unidade indica a velocidade da transmissão de dados, bits por segundo.

Figura 107 – Esquema de ligação e funcionamento do protocolo de comunicação UART.



Fonte: adaptado de (ROBOCORE, 2016).

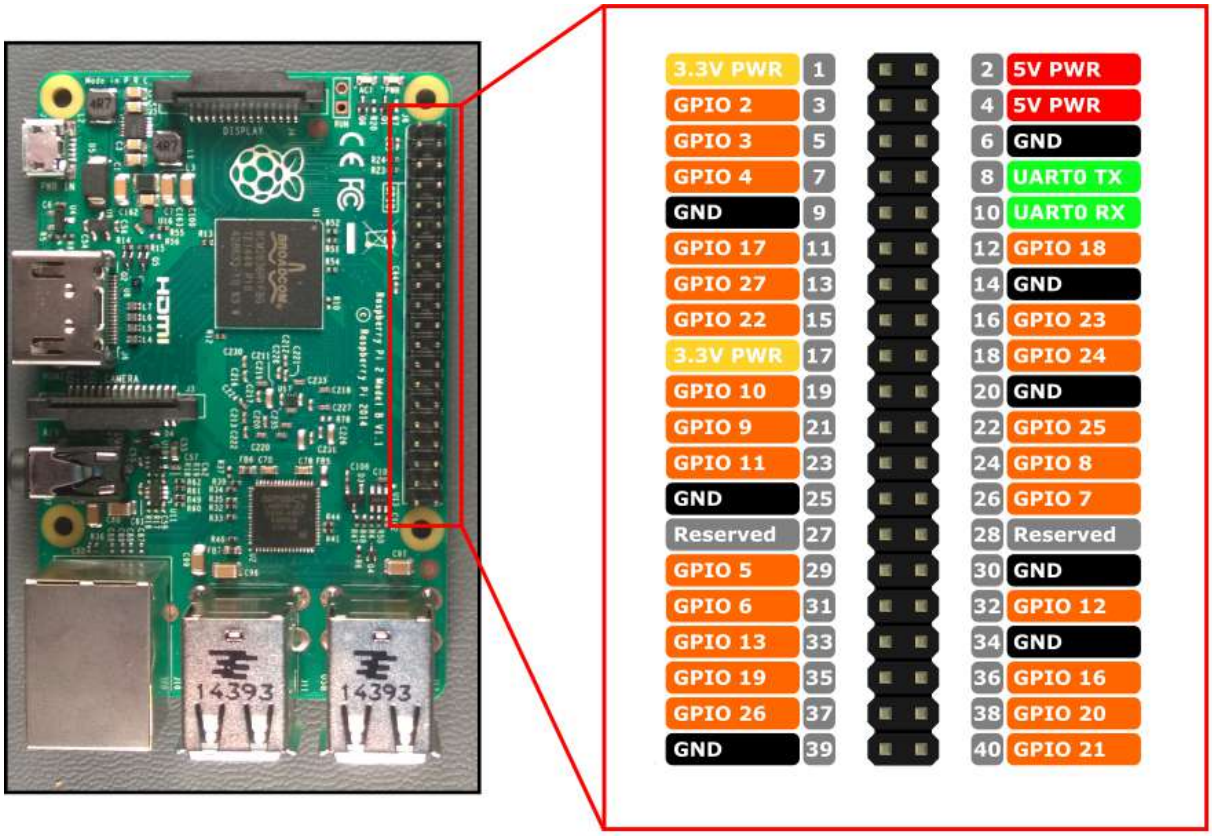
A partir da Figura 107, observa-se o modo de funcionamento do protocolo UART. No momento em que não existem dados a serem transmitidos, o barramento coloca-se em estado *Bus Idle*[5], mantendo-se em nível lógico alto. Por outro lado, quando existem dados a serem transmitidos, primeiramente é enviado um bit de inicialização (*start bit*), o qual está em nível lógico baixo. Logo após isso, um byte (*8 bits*) de informação é enviado. No processo de finalização do envio dos dados, envia-se um bit de parada (*stop bit*), o qual possui um nível lógico alto.

A compreensão do modo de funcionamento deste protocolo de comunicação é importante neste trabalho, pois na etapa do desenvolvimento dos algoritmos, deve-se levar em consideração a adequação da transmissão de informações entre os diferentes *hardwares*. No caso específico do uso da *Raspberry Pi*, a conexão física ocorre por meio dos pinos do *GPIO*[6].

A Figura 108 apresenta a especificação do GPIO da *Raspberry Pi 3*, utilizada no presente trabalho.

[5] Barramento inativo.
[6] Do inglês general purpose input/output ou entradas e saídas de uso geral.

Figura 108 – Raspberry Pi 3 GPIO.



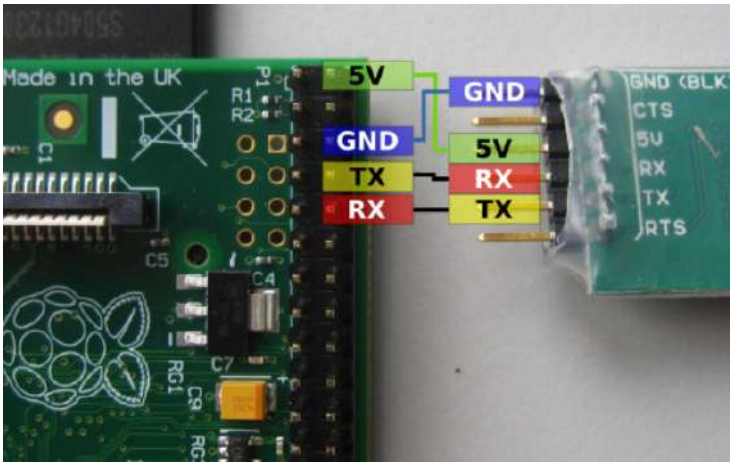
Fonte: próprio autor.

A partir da observação da Figura 108, nota-se que o pino 8 funciona como a porta transmissora (TX) e o pino 10 como a porta receptora (RX). De modo a habilitar a comunicação serial com protocolo UART na *Raspberry Pi*, faz-se necessário configurar os pinos oito e dez do *GPIO*.

O procedimento para a habilitação da comunicação serial na *Raspberry Pi* é realizado por meio da habilitação do modo UART no painel da configuração da própria *Raspberry Pi*, bem como algumas alterações em arquivos específicos do próprio sistema operacional (*Raspbian*). Uma vez realizadas as alterações necessárias, pode-se então realizar a comunicação serial da *Raspberry Pi*.

A Figura 109 apresenta um exemplo de ligação dos pinos oito e dez da GPIO para a comunicação UART da *Raspberry Pi* com outro dispositivo.

Figura 109 – Conexão serial UART da *Raspberry Pi*.

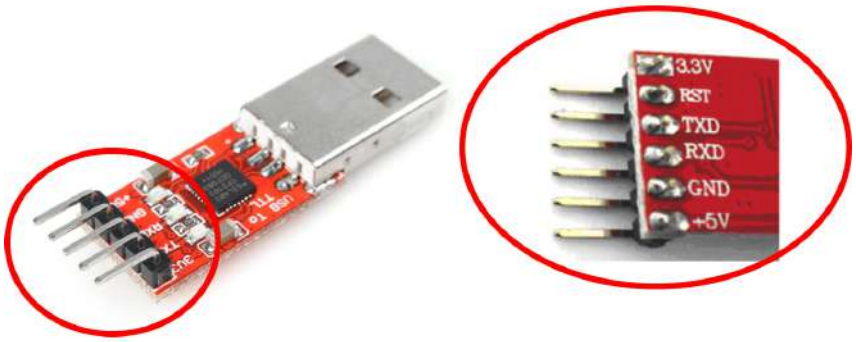


Fonte: (EICKHOLD, 2012).

No caso deste trabalho, utilizou-se um módulo conversor *USB*[7] para serial, permitindo dessa maneira realizar a comunicação entre a *Raspberry Pi* e o computador utilizado para o ambiente de simulação. Este conversor possui um chip *CP2102* integrado, o qual realiza a conversão *USB-UART* e funciona de modo similar à porta USB do computador por meio de um *driver COM*[8] virtual.

A Figura 110 apresenta o conversor USB-UART e suas conexões de comunicação.

Figura 110 – Módulo Conversor USB para Serial UART TTL CP2102.



Fonte: próprio autor.

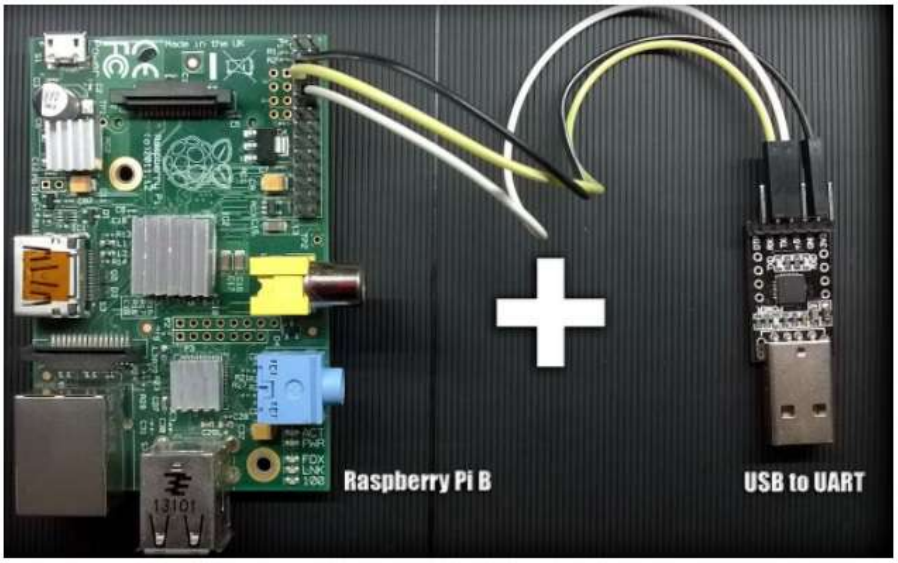
Uma vez finalizadas as configurações necessárias, o módulo conversor USB-UART é conectado à *Raspberry Pi*, obedecendo-se a pinagem de sua comunicação serial, ou seja, os

[7] Abreviação do inglês Universal Serial Bus ou Porta Serial Universal.
[8] Interface I/O que permite a conexão de um dispositivo serial.

pino oito e dez da *Raspberry Pi* são conectados aos pinos RX e TX do módulo conversor, respectivamente.

A Figura 111 apresenta um exemplo da conexão física da *Raspberry Pi* ao módulo conversor USB-UART.

Figura 111 – Simulação do controlador após o treinamento



Fonte: adaptado de (ESC, 2015).

Tabela 24 – Características do *Raspberry Pi* Modelo 3 B+.

SoC	Broadcom BCM2837B0 (CPU, GPU, DSP, SDRAM)
CPU	1.2 GHz ARMv8 Cortex-A53 Quad-Core (64bit)
GPU	Broadcom VideoCore IV H.264, MPEG-4 decode (1080p30); H.264 encode (1080p30); OpenGL ES 1.1, 2.0
SDRAM	1GB LPDDR2 (900 MHz)
USB	4 portas
Rede	2.4GHz e 5GHz IEEE 802.11.b/g/n/ac wireless LAN Bluetooth 4.2 Gigabit Ethernet over USB 2.0
Áudio	3.5 mm jack, HDMI e I2S áudio
Video	Painel LCD via DSI 14 resoluções possíveis HDMI (640x350-1920x1200) e diversos padrões PAL e NTSC
Alimentação	Fonte DC micro USB 5V/2.5A
Armazenamento	Cartão MicroSD
Consumo	500 mA
Peso	50g
Dimensões	85 x 56 x 17mm
Sistema Operacional	Raspbian, Ubuntu, Windows 10 IoT Core, Arch Linux ARM, SUSE 64 bits.