Brazilian Journal of Development

Proposta de uma ferramenta de ensino de inteligência artificial utilizando aprendizado por reforço aplicado a solução de labirintos dinâmicos

Proposal for a teaching tool of artificial intelligence using learning by enhancement applied to the solution of dynamic mazes

DOI:10.34117/bjdv6n3-248

Recebimento dos originais: 29/02/2020
Aceitação para publicação: 17/03/2020

Ênio dos Santos Silva
Professor do Departamento de Engenharia Elétrica do Instituto Federal de Santa Catarina – IFSC – Campus Itajaí (SC)
enio.silva@ifsc.edu.br

Douglas Hemkemaier da Silva
Aluno do Curso de Engenharia de Controle e Automação do Instituto Federal de Santa Catarina – IFSC – Campus Chapecó (SC)
douglas_hemkemaier@hotmail.com

Almir dos Santos Albuquerque
Doutorando do Programa de Pós Graduação em Engenharia e Gestão do Conhecimento – PPEGC
Universidade Federal de Santa Catarina - UFSC
asaalbuquerque@gmail.com

Rogério Cid Bastos
Professor Doutor do Programa de Pós Graduação em Engenharia e Gestão do Conhecimento – EGC
Universidade Federal de Santa Catarina - UFSC
rogerio@egc.ufsc.br

RESUMO

Este trabalho, apresenta o desenvolvimento de um sistema de inteligência artificial (IA) aplicado na navegação de robôs autônomos. Particularmente, o sistema de IA, aqui desenvolvido, é representado pela técnica de aprendizado por reforço (AR) aplicada para a solução de labirintos dinâmicos. A abordagem de diferentes áreas de pesquisa, como IA, processamento de sinais e controle e automação, permite a investigação de importantes temas da engenharia. Nesse contexto, este trabalho disponibiliza um framework de AR em robótica. Os resultados obtidos através das estratégias de AR, permitem inferir acerca da qualidade dos sistema de IA implementado e comprovam a eficácia do framework desenvolvido neste artigo.

Palavras-chave: Aprendizado por reforço (AR); Inteligência Artificial (IA); Robô Autônomo.

Brazilian Journal of Development

ABSTRACT

This research paper presents the development of an artificial intelligence (AI) system applied to the navigation of autonomous robots. Particularly, here the AI system is represented by the reinforcement learning (RL) technique applied to the solving of dynamic mazes. The approach of different research areas such as AI, signal processing and control and automation, allows the investigation of major issues of engineering. In this context, this work provides a RL framework in robotics. The results obtained through the RL strategies, allow to infer about the quality of the implemented AI system and prove the effectiveness of the framework developed in this article.

Keywords: Reinforcement learning (AR); Artificial Intelligence (AI); Autonomous Robot.

1 INTRODUÇÃO

Os avanços tecnológicos têm favorecido a implementação de ferramentas de apoio aos processos de ensino e apredizagem convencional, em diversas áreas das ciências. Então, a utilização do computador como instrumento de auxilio ao processo de ensino e apredizagem, tem se tornado uma essencial ferramenta didática e pedagógica, nos diversos sistemas educacioais do munto inteiro. Essas ferramentas, quando implementadas, utilizam diversas técnicas para atingirem seus objetivos. A inteligência artificial (IA) pode ser uma delas.

Russell (2014), define inteligência artificial (IA) como o conjunto de instruções inseridas em uma máquina (também nomeada de agente) a fim de transmitir alguma habilidade que simule a inteligência humana. Para a área da robótica, a IA consiste na investigação de agentes (inteligentes) que recebem sinais de percepção de um ambiente e executam determinadas ações de acordo com o sinal recebido, isto é, implementam funções que mapeiam sequências de percepções em ações (SILVA e SILVA, 2016). A aplicação de IA em robôs móveis é uma das áreas de grande destaque na robótica (SAKIB et al., 2014). Dentre as diversas aplicações disponíveis na literatura, encontram-se os robôs de resgate1, cujo objetivo é encontrar a vítima e identificar sua localização em meio a uma estrutura que simula um ambiente de desabamento.

Um problema semelhante, ao encontrado com robôs de resgates, é a solução de labirintos, considerado como um problema clássico quando se trata de robótica móvel. Segundo Coppin (2004), quando pessoas entram em labirintos, o mais comum é vagarem randomicamente em busca da saída, o que normalmente resulta em locomoverem-se em círculos. Uma possível solução para o problema do labirintos, é a “técnica da mão esquerda”, na qual a pessoa desloca-se mantendo a parede à esquerda como referência (RUSSELL, 2014).

[1]Modalidade presente em competições da *RoboCup Rescue Robot League* (http://www.robocup.org/)

Evidentemente, a aplicação da "técnica da mão esquerda" apresenta maior eficiência quando comparada à locomoção randômica. No entanto, ambas as soluções não utilizam, como referência, quaisquer informações adicionais de localização no labirinto durante o trajeto do agente até o destino desejado. Na literatura, a solução de labirintos pode ser obtida através de técnicas associadas à solução de problemas denominados grid world e de caminho mais curto (RUSSELL, 2014; GUPTA e SEHGAL, 2014). Nesse contexto, podem ser usadas as seguintes técnicas de IA: aprendizado por reforço (AR), programação dinâmica, redes neurais, algoritmos genéticos, entre outras (SAKIB et al., 2014; GUPTA e SEHGAL, 2014; TIJSMA et al., 2016). Particularmente, a busca por soluções inteligentes para o problema do labirinto ainda é um tópico ativo do estado da arte.

Neste trabalho de pesquisa, a aplicação de IA usando simulações de robôs autônomos é investigada. Especificamente, adota-se aqui a simulação de labirintos dinâmicos como ponto de partida para as investigações de técnicas de IA. Nesse contexto, uma estratégia usando AR é utilizada com base na experiência adquirida pelo robô a cada época de exploração em um dado labirinto. Assim, a solução ótima é obtida através do aprendizado do melhor caminho.

Espera-se com este trabalho, incentivar e fomentar as pesquisas sobre IA aplicadas em sistemas robóticos, facilitando as atividades dos que desejam atuar em IA, mas que atualmente devem vencer diversas barreiras para a configuração de um sistema básico. Acredita-se que, a partir deste trabalho de pesquisa, os estudantes sejam incentivados a projetar e construir robôs com habilidades de interação com o ambiente ao qual estão inseridos.

2 ROBOS AUTÔNOMOS

O desenvolvimento de um robô autônomo pode ser dividido em três estágios: percepção, planejamento e locomoção. No estágio de percepção, é realizada a coleta de informações através de sensores. Essas informações representam o ambiente externo que cerca o robô e são fundamentais para a sua orientação. No estágio de planejamento, os dados adquiridos são utilizados como parâmetros de referência para a realização do mapeamento do ambiente e da definição da ação a ser tomada. No terceiro estágio, o de locomoção, realiza-se a movimentação determinada no estágio de planejamento. Posteriormente, o ciclo retorna ao primeiro estágio, atualizando os dados adquiridos nas iterações anteriores (SILVA e SILVA, 2016).

O objetivo fundamental de um robô autônomo é ser capaz de construir um mapa e usá-lo para localizar-se no ambiente ao qual está inserido. Tal objetivo é comumente denotado

como localização e mapeamento simultâneos (simultaneous localization and mapping - SLAM). Um problema adicional importante é determinar se o robô está em uma parte do ambiente já armazenada ou nunca visitada. Nesse contexto, as estratégias determinísticas, como a navegação randômica e a "técnica da mão esquerda", não satisfazem os problemas de SLAM. Para contornar tal situação, faz-se necessário um planejamento aprimorado das trajetórias realizadas pelo robô, permitindo que o robô alcance o seu objetivo de maneira eficaz e eficiente.

De fato, a solução de labirintos pode ser obtida através de técnicas determinísticas, onde uma heurística predefinida conduz a locomoção do robô. Tal estratégia apenas obedece as instruções de uma programação estática (inflexível). No entanto, neste artigo, as técnicas de IA são utilizadas no estágio de planejamento de um robô autônomo. Dessa maneira, os robôs agem de acordo com as previsões provenientes do conhecimento adquirido do ambiente ao qual estão inseridos, sem nenhum conhecimento à priori. Particularmente, em IA o agente nem sempre parte de um objetivo conhecido e sequer conhece o ambiente ao qual está inserido, ele apenas interage e aprende com esse ambiente ao longo de suas experiências "vividas" (épocas). Para Coppin (2004), o aprendizado sempre tem a ver com o autoaprimoramento do comportamento futuro, baseado em experiências passadas.

A ideia de que podemos aprender através da interação com o meio ao qual estamos inseridos, é provavelmente a primeira ideia em mente quando pensamos sobre a natureza do aprendizado. Quando uma criança brinca, explorando o ambiante que a cerca, não há explicitamente um professor para ensinar-lhe como reagir ao ambiente, mas há uma conexão motora-sensorial direta com esse ambiente. Experimentar essa conexão (ligação) produz uma riqueza de informações importante sobre causa e efeito, sobre as consequências das ações, e sobre o que fazer, a fim de alcançar certos objetivos. Ao longo de nossas vidas, tais interações são, sem dúvida, uma fonte importante de conhecimento sobre o nosso ambiente e sobre nós mesmos. Aprender com a interação é uma ideia fundamental subjacente a quase todas as teorias de aprendizagem e inteligência (SUTTON e BARTO, 1998).

3 APRENDIZADO POR REFORÇO

Para a solução do problema de planejamento e SLAM, dois conceitos importantes são utilizados, a memória e o processo de decisão de Markov (PDM), (SUTTON e BARTO, 1998). A memória associa, essencialmente, as tentativas de solução do problema realizadas previamente, sendo elas falhas ou corretas. Assim, em processos futuros, ela pode ser utilizada

para a solução de problemas similares, baseando-se ou não em um modelo estimado (a cada época) do ambiente.

Particularmente, a utilização ou não de um modelo Mˆ estimado do ambiente é denominada model-based e model-free, respectivamente. Em ambos os casos, nenhum conhecimento à priori sobre o modelo é disponível. Dessa forma, o método model-based estima explicitamente o modelo a partir da experiência do agente usando algoritmos de programação dinâmica (SUTTON e BARTO, 998). Já o método model-free "aprende" a solução diretamente do ambiente (sem a necessidade de Mˆ), requerendo um custo computacional menor comparado ao método based-model, todavia podendo resultar em um aprendizado mais lento.

Considerando a estrutura de um labirinto, os obstáculos (paredes) e os corredores (espaços livres) são associados ao modelo real M do ambiente. Cada posição ("célula" do labirinto) é definida como um estado s possível e as ações a são definidas como a decisão do agente sobre qual será o seu estado futuro st+1 (posição futura).

Para a solução do cenário supracitado, contendo estados finitos e múltiplas ações, adota-se o conceito de PDM. Particularmente, a estratégia de aprendizado por reforço (AR) é formalmente descrita por um PDM. Assume-se que um processo estocático Y é considerado Markoviano se, e somente se, P(Yt+1 |Yt = i,Yt−1= kt−1,..., Y1 = k1, Y0 = k0) = P (Yt+1 = j | Yt = i), sendo o estado atual st estatisticamente suficiente para a estimação do estado futuro st+1.

No labirinto, cada estado s é considerado uma posição diferente do robô. E, em cada estado s é associado um valor de recompensa r que é diretamente relacionado com a obtenção do objetivo final do robô (que nesse caso é encontrar a saída do labirinto). Portanto, um PDM é um processo de recompensa de Markov dependente de decisões. Assim, todos os estados são Markovianos e a navegação no labirinto é dividida em estágios (ou passos). A cada passo, uma nova ação a é tomada considerando apenas o estado atual st.

Matematicamente, um PDM é definido como uma tupla (S, A, T, R, γ), onde:

S é o conjunto de estados finitos. O labirinto é dividido em células, quadradas de proporções iguais, e cada uma das células representa um estado diferente;

A é o conjunto de ações finitas. Para ocorrer a mudança de estado, o robô deve realizar uma ação de movimento, por exemplo, ir para a esquerda, frente ou direita;

T é a matriz de probabilidades de transição de estados T(st+1 | st, at). Probabilidade de ir à um estado futuro st+1, dado que no estado atual st será realizada a ação at;

Brazilian Journal of Development

R é a função de recompensas que atribui o valor da recompensa fornecida ao robô quando o estado s é alcançado, R(s, a) = E[r|s, a];

γ é um fator de desconto que indica a importância das recompensas futuras, γ ∈ [0, 1];

Dessa forma, o objetivo do agente é percorrer o ambiente e capturar o máximo (ou o mínimo) de recompensas possíveis até a chegada ao seu destino final, para isso, tal agente deve aprender uma política de aprendizado π que determine qual a ação ótima deve ser tomada em cada estado. A política π pode ser avaliada pela soma total das recompensas de um estado inicial s0 até o estado atual st. Essa soma é representada pela função valor Vπ(s) na Equação (1).

$$V^{\pi}(s) = r_t + \gamma r_{t-1} + \gamma^2 r_{t-2} + \ldots = E\{\sum_{k=0}^{\infty} \gamma^k r_{t-k} | s_0 = s, \pi\}$$
$$V^{\pi}(s) = \sum_{a_t} \pi(s_t, a_t) \sum_{s_{t+1} T(s_t, a_t, s_{t+1})} [R(s_t, a_t, s_{t+1}) + \gamma V^{\pi}(s_{t+1})] \quad (1)$$

Adicionalmente, a função $Q^*(s,a) = r(s,a) + \gamma \sum_{s_{t+1} \in S} T(s_t, a_t, s_{t+1}) V^*(s_{t+1})$, é definida para obter o valor atribuído a escolha da ação a no estado s, seguindo a política ótima π∗. Assim, em AR, a estimação da função Q∗ consiste no aprendizado de uma solução de um PDM.

Neste artigo, foram utilizados os algoritmos de AR denominados Q-learning e Dyna-Q, que são estratégias baseadas em free-model e based-model, respectivamente (SUTTON e BARTO, 1998; TIJSMA et al., 2016). Para a política de decisão π(s), adotou-se a estratégia ϵ-greedy apresentada na Equação (2), onde existe uma probabilidade ϵ de escolha aleatória da ação em cada estado, tal estratégia é mantida para que o agente sempre possa ter a possibilidade de explorar novos caminhos no ambiente. Caso ϵ = 0, o agente sempre escolherá a ação que retorna o máximo valor de Q(s, a).

$$\pi(s) = \begin{cases} a_{aleatório} & se \ \text{rand} < \epsilon, \\ \arg\max_a Q(s,a) & demais \ casos \end{cases} \quad (2)$$

Q-Learning

O algoritmo Q-learning é um método de aprendizado free-model, isto é, o processo de aprendizado independe da estimação de um modelo Mˆ do ambiente e é realizado interativamente a cada passo do agente, estimando, assim, continuamente a função Q para cada par estado-ação, de acordo com a Figura 1(a) e a Equação (3).

Brazilian Journal of Development

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + (1 - \alpha)[r_{t+1} + \gamma \arg\max_a Q_t(s_{t+1}, a_{t+1})] \quad (3)$$

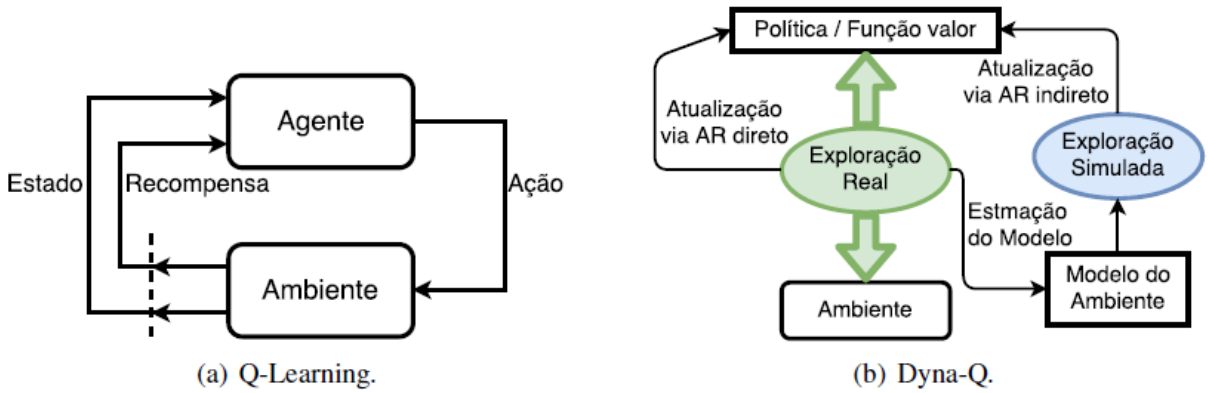
Onde α representa a taxa de aprendizado, indicando o quanto que as novas interações atualizam o que foi aprendido até o instante t.

Dyna-Q

O algoritmo Dyna-Q é um método de aprendizado model-based com simulação de ações. Nesse algoritmo, o aprendizado e as ações são adquiridos a partir, tanto da exploração do ambiente real, quanto da exploração de um ambiente simulado através de um modelo estimado Mˆ . A cada instante de exploração, novas informações são descobertas pelo agente e usadas para a estimação do modelo Mˆ do ambiente, permitindo um aprimoramento na política π de tomada de decisões.

Logo, para cada passo de exploração real no ambiente, existem dois estágios de aprendizagem. Esses estágios são denominados de model-learning e de AR direto, sendo usados para aprimorar a estimação do modelo Mˆ do ambiente (ou seja, torná-lo mais preciso e parecido com o ambiente real) e para aprimorar diretamente a função Q∗ como no algoritmo Q-learning. As relações entre a exploração do ambiente, o modelo Mˆ e a função Q∗ são apresentadas na Figura 1(b). Note que a exploração do ambiente pode aprimorar a estimação da função Q∗, tanto diretamente (AR direto) quanto indiretamente (AR indireto), via o modelo estimado Mˆ do ambiente.

Figura 1: Estratégias utilizadas nos algoritmos



Fonte: Elaborada pelos autores

Brazilian Journal of Development

4 IMPLEMENTAÇÃO DOS ALGORITMOS DE AR

Do quadro a seguir, nota-se que o método free-model Q-learning abrange as linhas de 1 a 6 do algoritmo. Já o método based-model Dyna-Q contempla todas as linhas do algoritmo, com destaque especial para a linha 7, representando a etapa de model-learning, e as linhas de 8 a 13 que implementam as etapas do planejamento (simulação) de n ações.

que implementam as etapas do planejamento (simulac¸a˜o) de n ac¸o˜es.

Algoritmo 1: Q-LEARNIG (LINHAS DE 1 A 6) E DYNA-Q (TODAS AS LINHAS)

1 **para** *épocas* $= 1$ *até* N_{epocas} **faça**
2 **para** *passos* $= 1$ *até* N_{passos} *(Exploração Real)* **faça**
3 $s_t \leftarrow$ o estado atual
4 $a_t \leftarrow$ a ação tomada para ϵ-$greedy(s_t, Q_t)$
5 Executa a_t, obtém a recompensa imediata r_{t+1} e o estado futuro s_{t+1}
6 $Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + (1 - \alpha)[r_{t+1} + \gamma \arg_{a_{t+1}} \max Q_t(s_{t+1}, a_{t+1})]$
7 $\hat{M} \leftarrow r_{t+1}, s_{t+1}$ (Estimação do Modelo)
8 **para** *simulação* $= 1$ *até* $n_{simuladas}$ *(Exploração Simulada)* **faça**
9 $s_t \leftarrow$ aleatoriamente um estado previamente observado
10 $a_t \leftarrow$ a ação previamente tomada em s_t
11 $r_{t+1}, s_{t+1} \leftarrow \hat{M}$
12 $Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + (1 - \alpha)[r_{t+1} + \gamma \arg_{a_{t+1}} \max Q_t(s_{t+1}, a_{t+1})]$
13 **fim**
14 **fim**
15 **fim**

É dessa forma, as estratégias discutidas na Seção 3 são implementadas.

5 RESULTADOS E ANÁLISE DE DESEMPENHO

Para a avaliação do desempenho das técnicas supracitadas, os robôs autônomos e os labirintos dinâmicos foram simulados usando o software Octave2. Especificamente, sensores de obstáculos foram adotados para o estágio de percepção. Tais sensores são inseridos nos robôs em suas posições frontal e lateral, permitindo a leitura de obstáculos (paredes) ou espaços livres (corredores) a frente, a direita ou a esquerda do robô. Dessa forma, as possíveis ações a executadas em um estado s (posição no labirinto) são: seguir em frente, retornar, dobrar a direita e dobrar a esquerda.

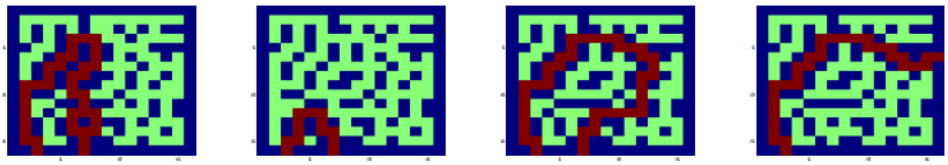
Os labirintos utilizados são de dimensão 16 x 16 e sofrem alterações ao longo do tempo (a cada 125 épocas) para simular a dinâmica dos labirintos, onde os melhores trajetos e os

[2] Octave é uma linguagem computacional, desenvolvida para computação matemática.

Brazilian Journal of Development

objetivos finais (destinos de saída, isto é, estado final sf) são periodicamente modificados. A Figura 2 apresenta as diferentes configurações dos labirintos, destacando especialmente os trajetos ótimos. A partir do labirinto ilustrado na Figura 2(a), o robô deve ser capaz de encontrar a saída sf , sempre adaptando-se para percorrer o trajeto mais curto (melhor política $\pi*$).

Figura 2: Configurações dos labirintos.



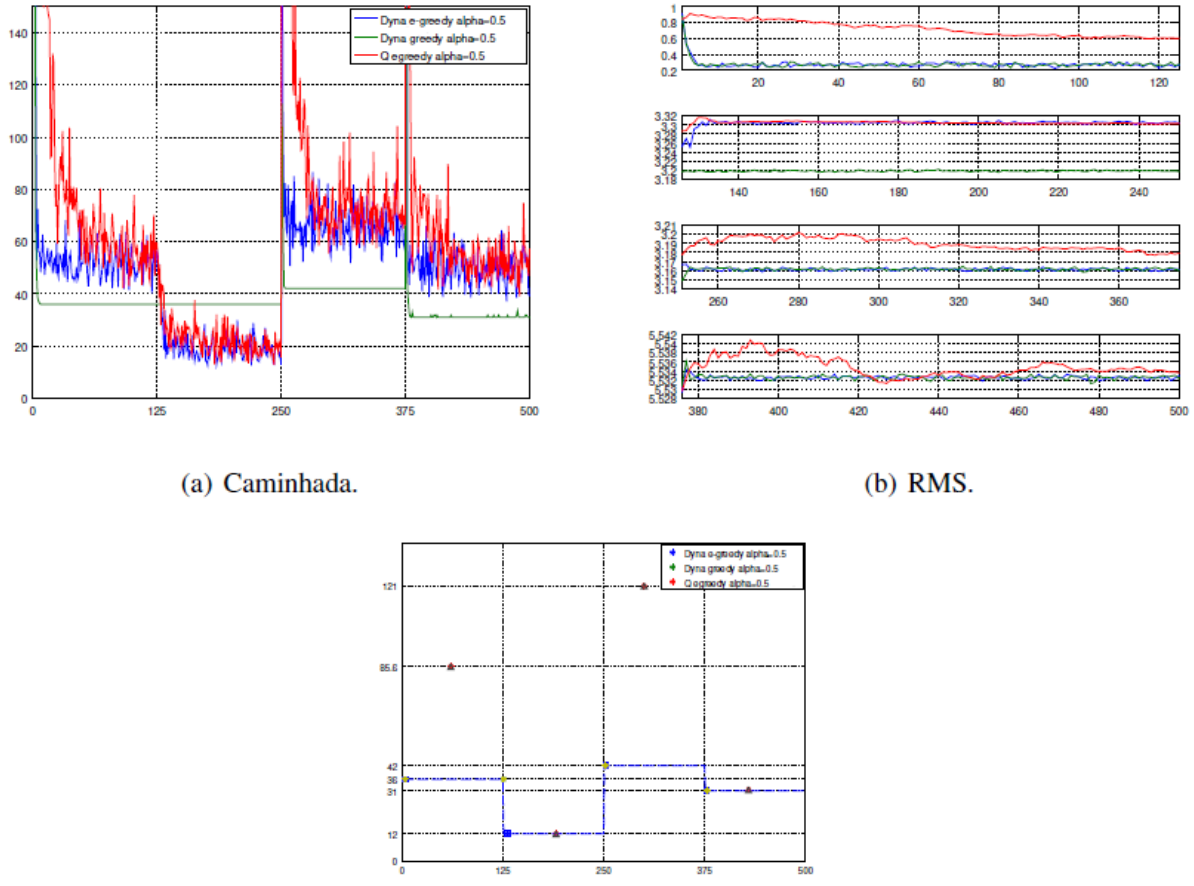
(a) Trajeto 1 (b) Trajeto 2 (c) Trajeto 3 (d) Destino 2

Fonte: Elaborada pelos autores.

Dessa forma, as estratégias discutidas na Seção 3 são implementadas. A Figura 3 apresenta os resultados das estratégias Q-learning greedy, Q-learning ϵ-greedy, Dyna-Q greedy e Dyna-Q ϵ-greedy. Aqui, foram usados os parâmetros $\alpha = 0,5$ (taxa de aprendizagem) e $\epsilon = 0,4$ (probabilidade de exploração aleatória). Nota-se da Figura 3(a) que a estratégia Dyna-Q ϵ-greedy apresenta os melhores resultados, obtendo uma convergência mais rápida e garantindo a adaptação para o melhor percurso (política $\pi*$). A Figura 3(b) apresenta as curvas da raiz do erro quadrático médio (Root Mean Square – RMS), entre as políticas de cada estratégia com a política ótima de cada labirinto. A Figura 3(c) apresenta a época e o melhor percurso de cada estratégia para cada configuração do labirinto.

Brazilian Journal of Development

Figura 3: Resultados.



(a) Caminhada. (b) RMS.

(c) Politica.

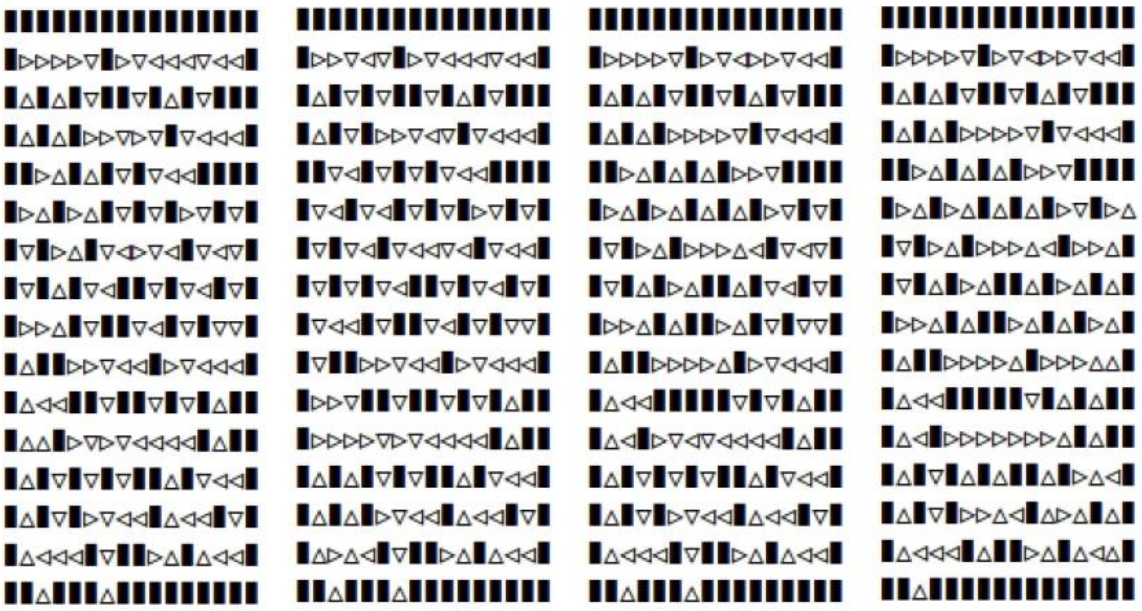
Fonte: Elabora pelos autores.

Ressalta-se que os algoritmos de AR obtêm resultados satisfatórios sem a necessidade de um conhecimento prévio sobre o labirinto. Em situações como a de resgate, o destino pode ser representado inicialmente pela localização de uma vítima e posteriormente pela localização de uma saída do labirinto. Nota-se que, nesses casos, as estratégias determinísticas são passíveis de falha, visto que não sabe-se a localização da vítima e, em caso de desmoronamentos, não podemos precisar a localização da saída bem como a estrutura interna do labirinto. Tal exemplo, evidencia a importância da capacidade do agente em adaptar-se às modificações que o labirinto (ambiente) possa vir a sofrer.

A Figura 4 apresenta as políticas resultantes do algoritmo Dyna-Q ϵ-greedy para as diferentes configurações do labirinto, onde as ações “cima”, “baixo”, “direita” e “esquerda”,

são representadas pelos símbolos “▵”, “▿”, “◃” e “▹”, respectivamente, e as paredes são representadas pelo síımbolo “ ▌”.

Figura 4: Políticas resultantes do algoritmo Dyna-Q ϵ-greedy.



Fonte: Elaborada pelos autores.

6 CONCLUSÃO

A técnica de AR aplicada para a solução de labirintos dinâmicos, oferece condições consistentes para servirem de base à construção e implementação da ferramenta proposta. A ferramenta a ser implementada baseada em robótica, se configurará como um grande avanço no ensino/aprendizagem de técnicas de IA, uma vez que utiliza os algoritmos de AR (Q-learning e, Dyna-Q). Especificamente, foi utilizado a simulação de um agente robótico com AR para a solução de labirintos dinâmicos.

Visando o incentivo de novos trabalhos de pesquisa na área de IA e robótica, os fundamentos teóricos de AR, bem como as principais técnicas de model-based e model-free foram apresentadas. Dessa forma, espera-se que a aplicação de sistemas de AR em robôs móveis auxiliem estudantes no processo de aprendizado de diversas técnicas relacionadas a IA e robótica.

Finalmente, os resultados das avaliações objetivas apresentaram as trajetórias ótimas e a capacidade de adaptação dos agentes robóticos aos labirintos, confirmando assim o desempenho satisfatório do framework disponibilizado neste trabalho de pesquisa.

REFERÊNCIAS

COPIN, B. Artificial Intelligence Illuminated. Jones & Bartlett Learning. 2004.

GUPTA, B.; SEHGAL, S. Survey on techniques used in autonomous maze solving robot. In Proc. of 5th Int. Conf. on Confluence The Next Generation Information Technology Summit, pages 323–328, 2014, Noida, India.

RUSSELL, P. N. S. Intelige^ncia Artificial. 3a Edição. Elsevier Brasil. 2014.

SAKIB, S.; CHOWDHURY, A.; AHAMED, S. T.; HASSAN, S. I. Maze solving Algorithm for line following robot and derivation of linear path distance from nonlinear path. 2014.

SILVA, E.; SILVA, D. Desenvolvimento de um sistema robótico controlado por reconhecimento automático de fala com adaptação ao locutor. In Proc. Of the 12th IEEE/IAS Int. Conf. on Industry Applications (INDUSCON), pages 2258–2266, 2016, Curitiba, PR, Brasil.

SUTTON, R.; BARTO, A. Reinforcement Learning: An Introduction. Bradford Book. 1998.

TIJSMA, A. D.; DRUGAN, M. M.; WIERING, M. A. Comparing exploration strategies for q-learning in random stochastic mazes. In Proc. of the IEEE Symposium Series on Computational Intelligence (SSCI), pages 351–359, 2016, Athens, Greece.