ROBÓTICA E PROGRAMAÇÃO: O USO DAS MATRIZES COMO TÉCNICA INTEGRADORA

Abraão Borges P. Azevedo (IFPB, Campus Esperança), Suemilton Nunes Gervazio (IFPB, Campus Esperança)

E-mails: abraao.borges@academico.ifpb.edu.br, suemilton.gervazio@ifpb.edu.br

Área de conhecimento (Tabela CNPq): 1.03.02.02-6 Modelos Analíticos e de Simulação

Palavras-chave: algoritmos; matrizes; robótica; bfs; aprendizagem baseada em problemas.

1. Introdução

O ensino nas áreas de Tecnologia da Informação enfrenta o desafio constante de tornar conceitos matemáticos e de engenharia mais significativos e contextualizados. Nesse sentido, a robótica desponta como uma abordagem eficaz, ao integrar programação, lógica computacional e fundamentos matemáticos, promovendo uma aprendizagem prática e interdisciplinar (Russell; Norvig, 2013). Além de desenvolver competências técnicas, a robótica educacional estimula habilidades como a resolução de problemas, o pensamento algorítmico e o raciocínio espacial, tornando-se uma ferramenta valiosa no processo de ensino-aprendizagem.

Entre os diversos recursos utilizados no desenvolvimento de sistemas robóticos, as matrizes assumem um papel central, especialmente no planejamento e execução de movimentos. Ao representar o ambiente como uma matriz bidimensional, é possível modelar com precisão os espaços livres, obstáculos e objetivos, fornecendo ao robô uma estrutura organizada para tomar decisões de deslocamento. As operações de movimentação, como avanços, desvios e rotações, podem ser interpretadas como transições entre elementos da matriz, possibilitando a implementação de algoritmos eficientes de navegação, como o Breadth-First Search (BFS) e a distância de Manhattan. Assim, este trabalho objetiva explorar o uso das matrizes como estrutura de dados fundamental na simulação de um robô aspirador, demonstrando como elas orientam o movimento e viabilizam a aplicação prática de conceitos matemáticos e computacionais na robótica educacional.

2. Materiais e métodos

A metodologia adotada consiste na elaboração de uma simulação computacional utilizando a linguagem Python e a biblioteca gráfica Pygame. O ambiente foi modelado como uma matriz bidimensional, representando espaços livres, obstáculos e áreas exploradas, enquanto o controle de navegação do robô é realizado por meio do algoritmo de busca em largura (Breadth-First Search – BFS) e a heurística da distância de Manhattan. O uso de heurísticas clássicas na programação é bem discutido por Pearl (1984), complementando a abordagem adotada.

A interface gráfica permite a visualização dinâmica das ações do robô, bem como a interação do usuário com a configuração do ambiente. Espera-se que essa abordagem possibilite a validação da representação matricial e dos algoritmos de navegação na simulação robótica, contribuindo para o processo de ensino-aprendizagem.

2.1 Modelo de equação

A Equação 1 calcula a distância de Manhattan entre dois pontos do ambiente:

$$D = |x_1 - x_2| + |y_1 - y_2|$$

em que D é a distância de Manhattan, e (x1,y1) e (x2,y2) são as coordenadas dos pontos no ambiente.

2.2 Algoritmo de Busca em Largura (BFS)

O BFS é empregado para explorar o ambiente e planejar rotas entre pontos. Trata-se de um algoritmo clássico de grafos, que explora o espaço a partir da posição inicial, expandindo sistematicamente para todas as células vizinhas acessíveis. O BFS utiliza uma fila (estrutura FIFO) para garantir que o caminho encontrado entre dois pontos seja o mais curto em número de passos, desde que não existam pesos diferenciados no ambiente (La Valle, 2006).

Esse algoritmo foi escolhido por sua simplicidade, eficiência e previsibilidade, características essenciais para ambientes discretos como a matriz da simulação.

3. Resultados e discussão

A simulação desenvolvida demonstrou a eficácia da matriz bidimensional como estrutura para representar ambientes em tarefas de exploração e limpeza robótica. O robô foi capaz de realizar a exploração completa do ambiente utilizando o algoritmo BFS, garantindo a cobertura sistemática de todas as células acessíveis e evitando repetições desnecessárias.

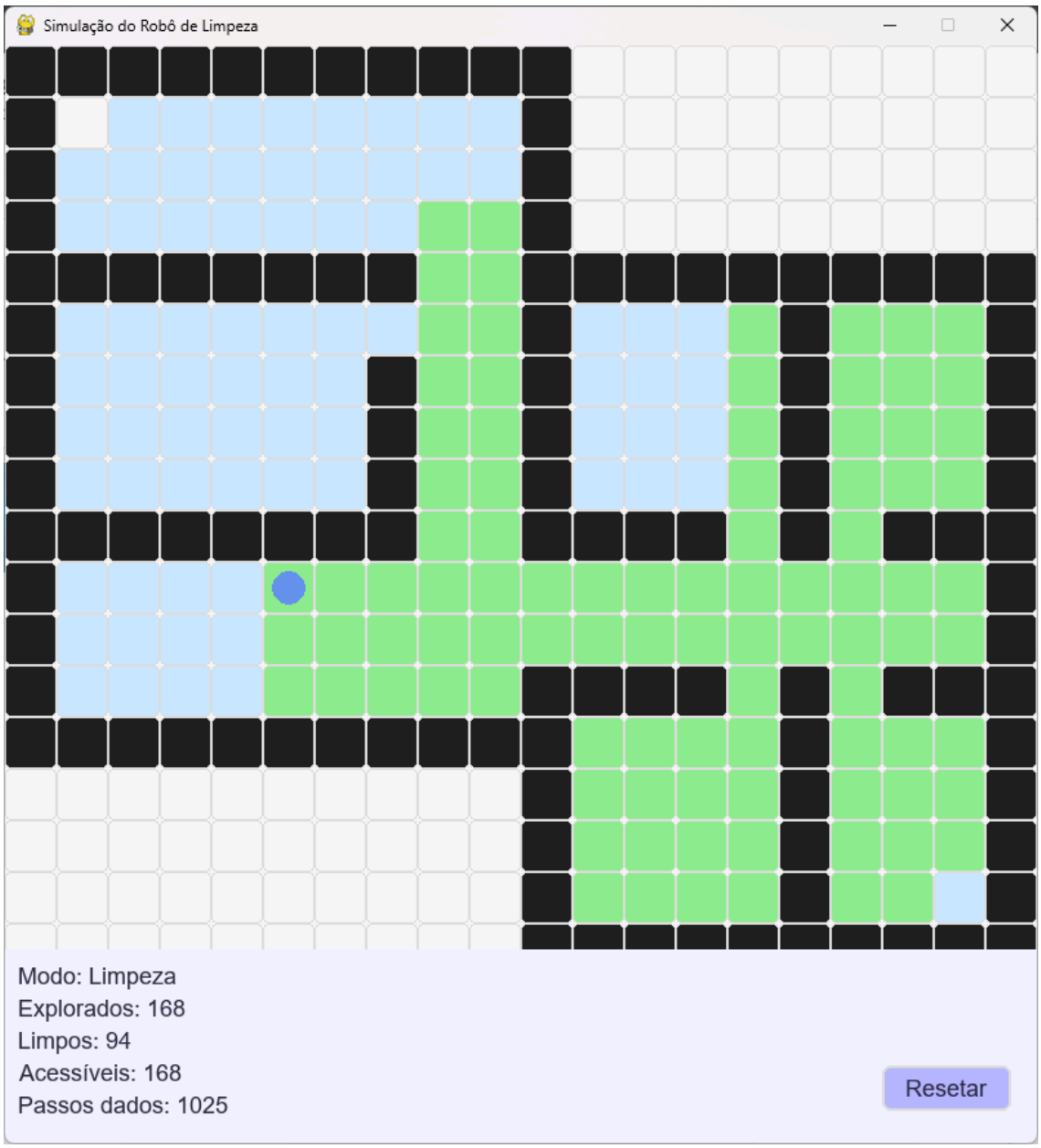
Durante a fase de limpeza, a aplicação da heurística da distância de Manhattan permitiu otimizar a sequência de deslocamento entre os pontos não limpos, resultando em trajetos eficientes e redução no número total de passos.

A interface gráfica, implementada com Pygame, proporcionou uma visualização clara e interativa da simulação, destacando estados das células (livres, exploradas, limpas, obstáculos) e a posição do robô em tempo real. Estatísticas como o número de células exploradas e limpas, bem como o total de movimentos realizados, foram exibidas dinamicamente, contribuindo para a análise e validação do comportamento algorítmico. A biblioteca Pygame é amplamente utilizada para esse tipo de aplicação (Pygame, 2025).

3.1 Modelo de figuras

A Figura 1 ilustra a representação gráfica do ambiente durante a simulação.

Figura 1 – Representação gráfica do ambiente simulado.



Fonte: Elaborado pelos autores (2025).

3.2. Tabelas

A Tabela 1 resume os estados e as cores representadas na matriz utilizada na simulação.

Tabela 1 – Estrutura da matriz utilizada na simulação.

Estado da célula	Representação	Cor
Espaço livre	vazio	branco
Obstáculo	parede	preto
Área explorada	explorado	azul claro
Área limpa	limpo	verde
Posição do robô	robô	azul escuro

Fonte: Elaborado pelos autores (2025).

4. Considerações finais

Com a implementação da simulação concluída, o trabalho evidenciou o potencial das matrizes como técnica integradora no ensino de robótica e programação, especialmente para representar ambientes e estruturar algoritmos de navegação. A expectativa é que a ferramenta desenvolvida sirva como estudo de caso para ilustrar a aplicação de conceitos matemáticos e computacionais em contextos práticos, fortalecendo o processo de aprendizagem. Futuramente, pretende-se expandir a simulação com novos algoritmos e funcionalidades, ampliando seu potencial educativo e experimental.

Agradecimentos

O presente trabalho conta com o apoio do Instituto Federal da Paraíba (IFPB), que fornece o ambiente acadêmico e os recursos necessários para o desenvolvimento desta pesquisa.

Referências

LA VALLE, S. M. Planning Algorithms. Cambridge: Cambridge University Press, 2006. Disponível em: http://planning.cs.uiuc.edu/. Acesso em: 23 maio 2025.

PEARL, J. Heuristics: Intelligent Search Strategies for Computer Problem Solving. Boston: Addison-Wesley, 1984.

RUSSELL, S.; NORVIG, P. Artificial Intelligence: A Modern Approach. 3. ed. Upper Saddle River: Pearson Education, 2013.

PYGAME. Pygame Documentation. Disponível em: https://www.pygame.org/docs/. Acesso em: 23 maio 2025.