Desenvolvimento de Software para Aplicações de Robótica Móvel

Software Development for Mobile Robotics Applications

Paulo Roberto Machado Silva Junior*, Márcio Rodrigues da Cunha†,
Andre Luiz Regis Monteiro‡, Pedro Victor Fontoura Zawadniak §,

RESUMO

A robótica é amplamente aplicada em diversas áreas. Dentre estas, no ensino, ela é utilizada em competições que buscam solucionar labirintos, onde um robô, chamado Micromouse, navega partindo de um ponto inicial e no menor tempo possível tenta atingir uma meta, normalmente, no centro do labirinto. Este trabalho buscou analisar duas técnicas de solução de labirintos, Flood Fill e Backtracking, apresentando, para diferentes tamanhos de labirintos uma análise gráfica: (I) da taxa de processamento, medida em segundos, (II) a quantidade de iterações que ambas técnicas realizaram até encontrar o objetivo. Através da análise gráfica, foi possível compreender que o Flood Fill se mostrou mais eficiente em relação ao número de iterações, porém, apresentando alta taxa de processamento. Por outro lado, o Backtracking apresentou baixo tempo de processamento e elevado número de iterações até alcançar a meta estabelecida. Portanto, o presente trabalho auxiliará na implementação de um robô Micromouse e a escolha de técnicas para solução dos labirintos, pois, com os resultados é permitido realizar uma estimativa do tempo de processamento e de quantas iterações a técnica levará para encontrar a solução.

Palavras-chave: Backtracking, Flood Fill, Micromouse.

ABSTRACT

Robotics is widely applied in several areas. Among these, in teaching, it's used in competitions that seek to solve mazes, where a robot, called Micromouse, navigates from a starting point and in a possible shortest time tries to reach a goal, usually in the center of the maze. This work sought analyze two techniques for solve maze, Flood Fill and Backtracking, showing a graphical analysis for different mazes size: (I) the processing rate, measured in seconds, (II) the number of iterations that both techniques performed until the objective. Through the graphical analysis, it's possible to understand that the Flood Fill was more efficient in relation to the number of iterations, however, presenting high processing hate. On the other hand, Backtracking presented low processing time and high number of iterations until reaching the established goal. Therefore, the present work will help in the implementation of a Micromouse and the choice of techniques for solving the mazes, with the results, it's possible to estimate the processing time and how many iterations the technique will take to find the solution.

Keywords: Backtracking, Flood Fill, Micromouse.

* Engenharia Eletrônica, Universidade Tecnológica Federal do Paraná, Campo Mourão, Paraná, Brasil; pjunior.2018@alunos.utfpr.edu.br
† Universidade Tecnológica Federal do Paraná, Campo Mourão, Paraná, Brasil; prof.marciocunha@gmail.com
‡ Universidade Tecnológica Federal do Paraná, Campo Mourão, Paraná, Brasil; almonteiro@utfpr.edu.br
§ Universidade Tecnológica Federal do Paraná, Campo Mourão, Paraná, Brasil; zawadniak@alunos.utfpr.edu.br

1 INTRODUÇÃO

A robótica se caracteriza por ser constituída pela convergência de ciência, de engenharia e de tecnologia, visando a criação e a produção de máquinas, conhecidas como robôs. Segundo (MARTINS, 2016), a robótica é tida como a ciência dos sistemas que interagem com o mundo real, com pouco ou mesmo nenhuma intervenção humana.

O processo de robotização, passa por um período de grande disseminação em diversos setores do arranjo produtivo global. De acordo com (CERQUEIRA; FRANCISCO, 20XX), a globalização é responsável por diversas modificações no mundo. Um desses fatores, fortemente influenciados pelo processo de globalização, foi o surgimento da robotização nos processos industriais e o desenvolvimento de algoritmos capazes de realizar a integração hardware e software.

A robótica engloba diversos ramos, articulando uma alta complexidade, pois, envolve sistemas com diversas partes mecânicas, eletromecânicas e, muitas vezes, controlados por diversos Circuitos Integrados (CIs). Com os CIs é possível transformar pequenas partes eletromecânicas em conjuntos complexos, sendo capaz de extrair informações do meio em que se encontra (MARCULA, 2019).

A robotização de um processo se dá de maneira mais fácil, quando este é um processo repetitivo (simples) e que não envolve diversas variáveis. Contudo, a robótica está presente em diversos ramos, tais como: educação; indústria; medicina; tecnologia espacial (LEAL, 2003).

Segundo (SILVA, 2016), particularmente no ramo da educação e da tecnologia, em 1977, quando era editor chefe do Instituto de Engenheiros Eletricistas e Eletrônicos (IEEE), Donald Christiansen, propôs que fossem criadas competições, onde, os robôs deveriam ser completamente autônomos e que pudessem encontrar a solução de um problema proposto. Esses robôs criados receberam o nome de “Micromouse”. O problema proposto era alcançar a região central do labirinto, a partir de um predeterminado ponto de partida. A partir desse desafio, diversas técnicas de geração e de solução de labirintos foram criadas, dentre as de solução destaca-se o Flood Fill e Backtracking. Neste contexto, qual dessas técnicas apresenta melhor eficiência em solucionar labirintos? Qual apresenta menor tempo de processamento até encontrar a solução?

Nesta perspectiva, o Micromouse é um robô capaz de mapear e encontrar qualquer destino de um labirinto, desde que este seja preestabelecido, a partir de um algoritmo de solução de labirintos. Portanto, o objetivo deste trabalho é implementar as técnicas Flood Fill e Backtracking e realizar a comparação do desempenho computacional destas técnicas.

2 MÉTODO

O Micromouse se caracteriza por ser um robô móvel, que possui movimento realizado por motores acoplados nas rodas e fixados em seu chassi e utiliza distintos sensores, para que seja possível fazer a aplicação do mesmo em mapeamentos para solucionar labirintos. O Micromouse é capaz de obter informações do ambiente em que se encontra, através de sensores, e planejar suas próximas ações.

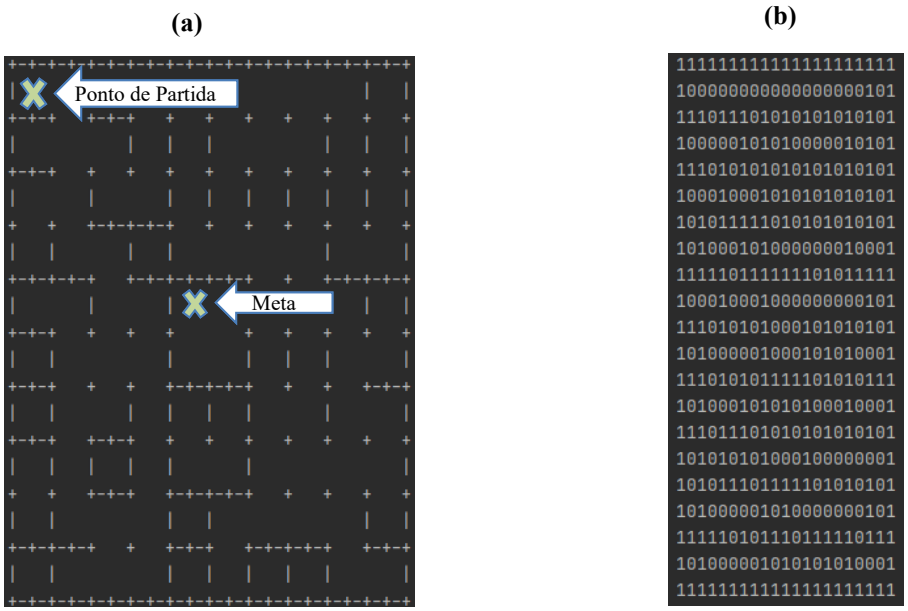
O controle do robô é feito por meio da integração de hardware e software, essa ação é realizada por um sistema capaz de processar os sinais de entrada, converter e processa-los para realizar alguma tarefa, de acordo com o software desenvolvido para solucionar os labirintos.

Logo, a pesquisa se iniciou com (1) a escolha e implementação de uma técnica de geração de labirintos, (2) implementação das técnicas Flood Fill e Backtracking, (3) teste das técnicas em labirintos de diferentes tamanhos, (4) coleta dos dados (tempo de processamento e número de passos) e (5) comparação das técnicas.

As pistas foram estruturadas a partir de rotina desenvolvida em Python para gerar labirintos aleatórios, que podem ser aplicados para geração de NxN tamanhos. A Figura 1 mostra o labirinto gerado e a matriz binária correspondente.

Figura 1 - Labirinto gerado e matriz binária correspondente.

(a) **(b)**



Fonte: Autoria Própria (2021).

A Figura 1.a traz um labirinto gerado através da rotina desenvolvida com o tamanho de 10x10. Para que as técnicas de solução sejam aplicadas, os labirintos são convertidos para matrizes binárias 2D, onde “1” representa a existência de alguma parede e “0” a ausência delas. A Figura 1.b traz uma matriz binária gerada a partir do labirinto presente na Figura 1.a.

Segundo (WILLARDSON, 2001), a técnica Flood Fill é um dos mais complexos meios de solucionar labirintos, pois, necessita de uma suposição inicial afirmando que não existem paredes no labirinto e, para cada célula é atribuído um valor. Esse valor atribuído, corresponde a quantidade de blocos percorridos desde o momento de partida até aquele determinado ponto.

Por outro lado, o método Backtracking baseia-se em encontrar uma solução através de um processo de repetição, a recursividade, consistindo em solucionar um bloco por iteração (101COMPUTING, 2017). Para soluções insatisfeitas, essa técnica retorna à posição anterior a que se encontra atualmente. A recursividade se dá através da função chamar ela mesma, porém, com parâmetros distintos dos anteriores.

3 RESULTADOS

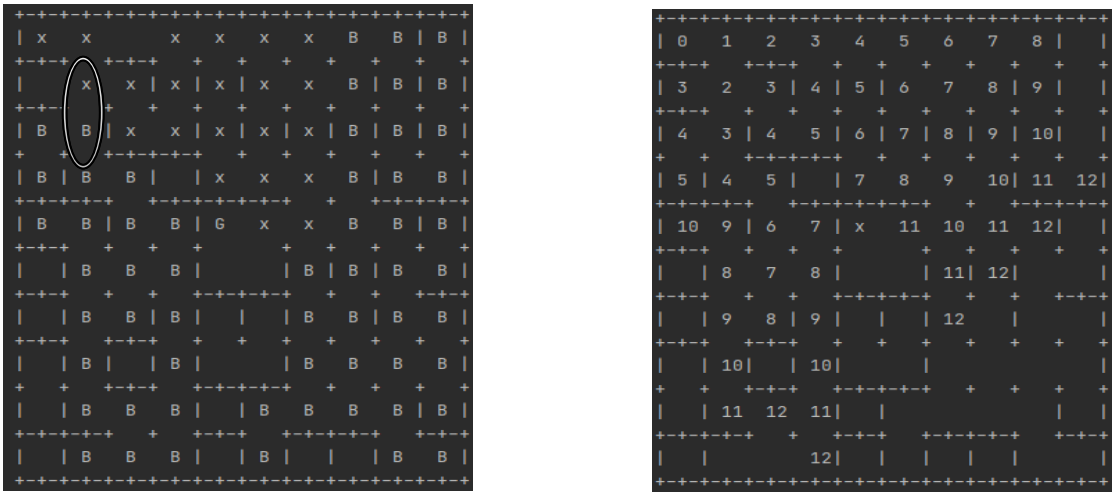
A partir da rotina de geração de labirintos, Figura 1, foram aplicadas as técnicas de solução Backtracking e Flood Fill, em diferentes tamanhos de labirintos. A condição de contorno inicial foi estabelecer o canto superior esquerdo como ponto de partida, ou seja, as coordenadas [1,1] da matriz correspondente ao labirinto, e a meta sendo o meio do labirinto.

Aplicando a técnica Backtracking, foi encontrado uma solução apresentada na Figura 2.a. O caractere “X” representa o caminho feito pelo algoritmo; “G” representa o objeto do algoritmo, onde ele precisa chegar a

partir das coordenadas iniciais; “B” corresponde ao ponto em que o algoritmo não encontrou uma saída para aquele trajeto, e precisou retroceder até um nó onde apresentava uma outra alternativa de caminho.

Figura 2 - Solução por Backtracking e por Flood Fill.

(a) **(b)**



Fonte: Autoria Própria (2021).

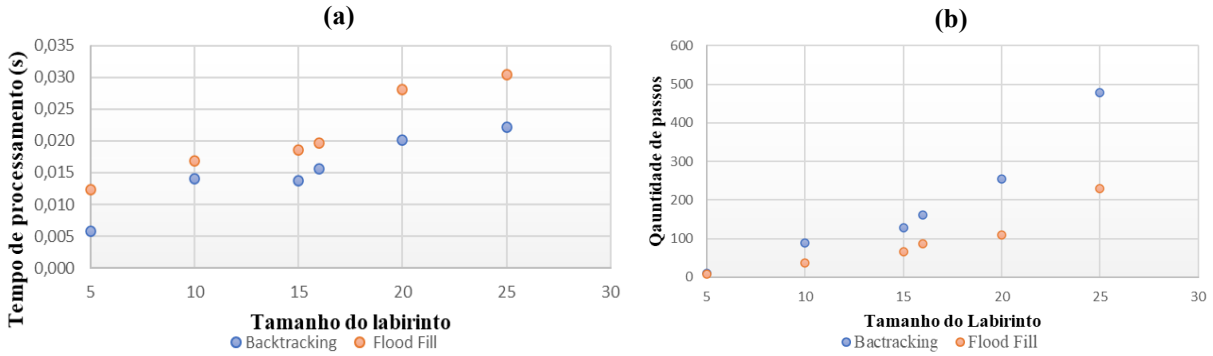
Analisando a solução obtida por meio da técnica Backtracking, de imediato é possível visualizar que, antes de chegar ao objetivo final, houve a necessidade de percorrer quase todo o labirinto. Até alcançar o objetivo preestabelecido, o algoritmo precisou dar 81 passos e levou um tempo de aproximadamente 4,575 ms. Sempre que se depara com uma bifurcação, ele escolhe uma das alternativas e segue até tentar encontrar uma saída. Esse tipo de algoritmo, muita das vezes, recebe o nome de algoritmo guloso, pois ele busca uma solução rápida, porém, não necessariamente a melhor, mas pode ser.

O algoritmo Flood Fill atribui valores para cada uma das células, onde esse valor representa a distância percorrida do ponto de partida até a presente célula. Aplicando a técnica Flood Fill ao labirinto apresentado pela Figura 1.a, que possui o mesmo objetivo estabelecido para a técnica Backtracking, obteve-se o resultado mostrado na Figura 2.b. Observe que, a posição de partida [1,1] possui o valor “0” porque o algoritmo não precisa realizar nenhuma iteração para chegar ao ponto de partida, já as demais posições, possuem um valor correspondente a distância percorrida.

Quando a técnica Flood Fill encontra as coordenadas que representam o objetivo, esse é preenchido com “X”, sendo assim, ele possui ao mínimo uma solução para o labirinto, que é expressa por um vetor de coordenadas matriciais. Até alcançar o objetivo, essa técnica precisou realizar 60 iterações em um tempo de aproximadamente 5,248 ms. Analisando a Figura 2.a e Figura 2.b, o Flood Fill não precisou analisar todo o labirinto, já que essa técnica apresenta uma análise mais eficiente a respeito do próximo passo.

Completando a caracterização dos dois métodos apresentados, foram realizados diversos testes com diferentes tamanhos de labirinto. Os resultados são apresentados na Figura 3.

Figura 3 - Tempo de processamento e quantidade de passos em relação ao tamanho do labirinto.



Fonte: Autoria Própria (2021).

O grau de complexidade é atrelado proporcionalmente ao tamanho do labirinto. Analisando a Figura 3.b, conforme vai aumentando o tamanho, mais caminhos começam a surgir e mais escolhas o algoritmo precisará tomar, consequentemente, aumentando o tempo de processamento de ambos métodos. Assim como o tempo de processamento, a quantidade de passos que o algoritmo usará para encontrar a solução para o labirinto, é atrelada diretamente ao tamanho do mesmo.

Pela Figura 3.a, é possível verificar que o tempo de processamento e a quantidade de passos, estão atrelados diretamente a complexidade do labirinto e o método utilizado para soluciona-lo; o Backtracking, por ser um método guloso, realiza mais passos até encontrar o resultado. Porém, por utilizar a recursividade, seu tempo de processamento é menor do que o Flood Fill; em média, o Flood Fill apresentou metade da quantidade de passos que o Backtracking, pois ele possui uma escolha de caminho mais inteligente.

De acordo com os resultados apresentados, frequentemente, a técnica Backtracking não retornará a melhor solução, pois se trata de um algoritmo guloso e seguidamente não consegue optar por rotas mais curtas e escolhem caminhos que levam para mais longe do objetivo, tornando assim a quantidade de passos superior ao Flood Fill. Como apresentado, existem diversas técnicas para solucionar labirintos, desde as mais fáceis (Backtracking), até as mais complexas (Flood Fill). Fazendo um estudo mais detalhado sobre essas duas técnicas, foi constado que o Flood Fill apresenta uma alta taxa de processamento, pois muitos números precisam ser atualizados e muitos kilobytes de RAM são necessários; o Backtracking é de fácil implementação, sempre alcançará o destino, porém, nem sempre navegará pelo melhor caminho.

4 CONCLUSÃO

Dentre os dois métodos escolhidos para solucionar labirintos, o Backtracking é um método básico, que frequentemente consegue encontrar uma solução, mas sua forma de solucionar não é a mais eficiente. O Flood Fill mostrou ser bastante eficiente em solucionar labirintos complicados, porém, com um tempo de processamento maior e quantidade de passos inferior, se comparado ao Backtracking.

Os resultados permitem realizar uma estimativa do tempo de processamento e de quantas iterações a técnica levará para encontrar a solução. Logo, na tarefa de implementação do robô Micromouse, a integração hardware/software terá uma estimativa com a aplicação da técnica e já ter previamente um resultado, sobre o tempo e a quantidade de passos de acordo com o tamanho do labirinto, ressaltando o intervalo modelado, ou seja, para labirintos com tamanho de 5 até 25.

Portanto, pode-se realizar trabalhos futuros voltados às aplicações práticas para ambas as técnicas apresentadas, em função da disponibilidade de hardware para implementação, podendo optar por baixo tempo de processamento e alta quantidade de passos, ou elevado tempo de processamento, porém, com baixa quantidade de passos.

REFERÊNCIAS

101COMPUTING. **Backtracking Maze – Path Finder**. 2017. Disponível em: https://www.101computing.net/backtracking-maze-path-finder/.

CERQUEIRA, W.; FRANCISCO. **A robotização na produção industrial**. Mundo Educação. 20XX. Disponível em: https://mundoeducacao.uol.com.br/geografia/a-robotizacao-na-producao-industrial.htm . Acesso em 12 jan. 2021.

MARTINS, A. **O que é Robótica**. São Paulo, Editora Brasiliense, 2006. **Micromouse Competition Rules**. IEEE. 2018. Disponível em: https://ewh.ieee.org/sb/columbus/devry/sacFiles/MicromouseRules.pdf. Acesso em 14 jan. 2021.

LEAL, R. D. G. **IMPACTOS SOCIAIS E ECONÔMICOS DA ROBOTIZAÇÃO: ESTUDO DE CASO DO PROJETO ROBOTURB**. UFSC. Disponível em: https://core.ac.uk/download/pdf/30383046.pdf. Acesso em 30 jul. 2021.

SILVA, K. L. **Micromouse: Um robô solucionador de labirinto**. Embarcados. 2016. Disponível em: https://www.embarcados.com.br/micromouse/. Acesso em 08 set. 2021.

TONÉIS, G. Y.; YAMASHITA, C. N. **A Cinemática de um Micromouse: Modelos matemáticos aplicados a um robô móvel**. IFG. 2020. Disponível em: https://www.academia.edu/43722937/A_Cinem%C3%A1tica_de_um_Micromouse_Modelos_matem%C3%A1ticos_aplicados_a_um_rob%C3%B4_m%C3%B3veL. Acesso em 11 jan. 2021.

WILLARDSON, D. M. **Analysis of Micromouse Maze Solving Algorithms**. Portland State University. 2001. Disponível em: http://web.cecs.pdx.edu/~edam/Reports/2001/DWillardson.pdf. Acesso em 13 jan. 2021.