UNIVERSIDADE ESTADUAL DE FEIRA DE SANTANA
Autorizada pelo Decreto Federal nº 77.496 de 27/04/76
Recredenciamento pelo Decreto nº17.228 de 25/11/2016
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
COORDENAÇÃO DE INICIAÇÃO CIENTÍFICA

XXV SEMINÁRIO DE INICIAÇÃO CIENTÍFICA DA UEFS
SEMANA NACIONAL DE CIÊNCIA E TECNOLOGIA - 2021

Desenvolvimento de um Sistema de Navegação Baseado no Algoritmo BUG

Gustavo Henrique Bastos de Oliveira[1] e Anfranserai Morais Dias[2]
1. Bolsista PIBIC/FAPESB, Graduando em Engenharia de Computação, Universidade Estadual de Feira de Santana, e-mail: ghboliveira@hotmail.com
2. Orientador, Departamento de Tecnologia, Universidade Estadual de Feira de Santana, e-mail: anfranserai@ecomp.uefs.br

PALAVRAS-CHAVE: Navegação; Algoritmo Bug; Marcos Fixos.

INTRODUÇÃO

A cada ano podemos observar a crescente utilização de robôs em diversos ambientes, tanto na indústria, quanto nas residências [1]. Essa democratização do acesso aos robôs se deve a diversos fatores, tanto econômicos quanto práticos [1][2]. A indústria percebe essa movimentação de interesses, criando soluções que atingem problemas específicos, fazendo com que novas tecnologias sejam construídas.

Com o objetivo de um menor custo computacional para embarcar em computadores com especificações mais simples, surgiram alguns algoritmos de navegação de baixo custo computacional. Dentre eles, existe uma classe de algoritmos chamada de algoritmos bug [3][4], que são capazes de criar trajetórias entre um ponto de origem e um ponto de destino, mesmo em ambientes onde os obstáculos não são previamente conhecidos. Isso oferece vantagens comparado a outros algoritmos de planejamento de rotas, pois não precisa de uma análise completa do ambiente no qual estão os pontos iniciais e finais. Cada algoritmo bug possui suas estratégias de reconhecimento de obstáculos e decisões de desvio, a escolha do algoritmo a ser utilizado vai centrar-se no desempenho na construção de um caminho válido entre os objetivos.

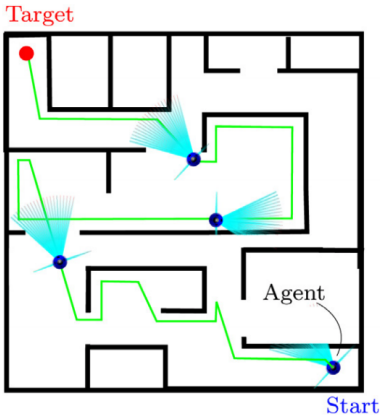
Traçar o caminho é apenas uma parte de um sistema de navegação, pois é necessário controlar motores, processar informações e tomar decisões. Para evitar a necessidade de construir todo esse sistema de controle, foi utilizado o ROS (*Robot Operating System*) que fornece conjuntos de sistemas robóticos que são responsáveis por esta gestão.

MATERIAL E MÉTODOS OU METODOLOGIA (ou equivalente)

No exercício de suas atividades, o robô deve ser capaz de compreender a sua posição no espaço, interpretando o ambiente ao redor. Isso é importante para que seja possível transitar neste ambiente, evitando possíveis obstáculos e alcançando posições desejadas. Uma das soluções para este problema são os algoritmos BUG, que são algoritmos de

navegação utilizados para em ambientes desconhecidos[5][6]. A figura 1 ilustra o funcionamento básico.

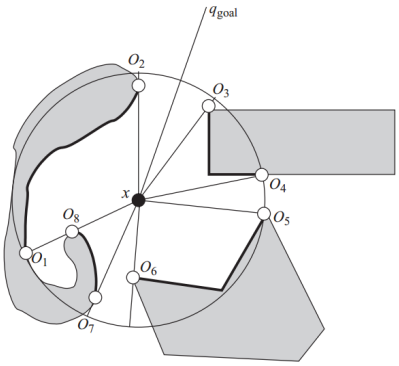
Figura 01: Exemplo de trajetória criada com um algoritmo BUG.



Fonte: McGuire, Kimberly N.[4]

Na imagem acima podemos observar um exemplo de trajetória criada utilizando um algoritmo BUG, com a origem e destino sinalizados em azul e vermelho, respectivamente. A primeira vez que o termo algoritmo BUG apareceu foi na década de 80 com o objetivo de solução de labirintos. Partindo da ideia de um labirinto com apenas uma entrada e saída, se o destino for alcançável a partir de paredes interconectadas, um seguidor de parede garantiria a solução uma solução mais rápida do que um caminhante aleatório[7]. Por conta desta característica de orientar-se pelas paredes, os algoritmos BUG eram considerados algoritmos de contato, com o tempo foram surgindo novos algoritmos baseados nessas idéias, adicionando sensores e tornando-os mais complexos e eficientes, isso fez que surgissem outras classes de algoritmos. Dentre esse conjunto de algoritmos, um dos que mais chamam a atenção são os TagentBug. Diferente dos primeiros algoritmos BUG, os TagentBug utilizam sensores de alcance, que devem informar a distância entre os obstáculos, não sendo necessário o contato com as paredes para calcular um desvio.

Figura 02: Trajetória criada pelo TangentBug.



Fonte: Utku Çulha.

Como podemos observar na figura 2, com os dados dos sensores, o algoritmo detecta os obstáculos e demarca pontos de descontinuidades(O), criando um grafo de visibilidade, onde são ligados os pontos de descontinuidades para representar um obstáculo, como na aresta constituída pelos pontos O_1 e O_2, a partir daí traçar segmentos de retas, tomando como opção a reta que leve-o para o mais próximo possível do destino cartesianamente calculado. Dividindo essa ação em pedaços, o algoritmo é capaz de chegar ao seu objetivo, repetindo-o em cada interação.

Na implementação do TangentBug, encontramos algumas implementações disponíveis na comunidade, no entanto, nenhuma totalmente compatível com o ROS e com o sistema de localização por marcos fixos. Então foi criado um pacote com a finalidade de utilizar esses sistemas e realizar a integração com a pilha de navegação do ROS, que é um controlador de velocidade e posição, no entanto, algumas adaptações tiveram que ser feitas, substituindo o uso de sensores a laser por sonar. A troca do sensor a laser por sonar foi feita por conta do robô utilizado, ele possui um conjunto de 16 sensores, 8 na parte frontal e 8 na parte traseira, a utilização do laser necessita de um processamento de dados mais complexo e caro computacionalmente. Esses sensores são fornecedores dos dados utilizados na criação do grafo de visibilidade.

Ao se trabalhar com robôs torna-se imprescindível a execução de testes em ambientes simulados, pois torna o trabalho de construção e verificação muito mais fácil. Para isso existem diversos programas de simulação como o RVIZ, Gazebo entre outros, compatíveis com o ROS[8]. Esses simuladores servem para avaliar a execução do conjunto completo funcionando, ou seja, tentam criar um ambiente completo, simulando efeitos físicos de um mundo real. Porém, antes disso, são realizados testes de unidade, para avaliar a execução de cada subsistema, que utilizam simulações mais simples.

RESULTADOS E/OU DISCUSSÃO (ou Análise e discussão dos resultados)

Para a construção de testes, foi criado um ambiente básico utilizando as ferramentas de simulação do ROS. Capturamos dos sensores alguns valores para realizar os testes. Ajustando em diferentes locais tanto os sensores, quanto a posição do robô. Na imagem abaixo podemos ver o mapa de teste criado para as simulações. Nele podemos variar as posições dos sensores e do robô.

Figura 05: Mapa das simulações.

Fonte: Próprio Autor.

CONSIDERAÇÕES FINAIS (ou Conclusão)

Por conta da finalização da bolsa, alguns dados relacionados com a execução do planejamento de rota e localização não foram devidamente obtidos e analisados. Existe a possibilidade de melhoria no sistema de localização mencionado e também no grafo de visibilidade dos algoritmos TangentBug. Ambas melhorias iriam melhorar o desempenho do sistema como um todo.

Os algoritmos BUG possuem uma eficiência x custo computacional melhores que os algoritmos tradicionais, pois ele baseia-se no método evasão de obstáculos, então não precisa de um mapa informando todos os detalhes do ambiente, apenas de sensores capazes de reconhecer a sua posição e a posição de obstáculos próximos, então o processamento de dados está centrado nessa pequena visão do ambiente que é obtido dos sensores. No entanto, podem até não construir o melhor caminho possível para um determinado ambiente, mas conseguem levar ao destino.

REFERÊNCIAS

[1] ZARUVNI, Reinaldo. Aspiradores robô tiveram aumento em vendas de 372% em 2020, 27 de nov. de 2020. Disponível em: <https://www.tecmundo.com.br/produto/207430-aspiradores-robo-tiveram-aumento-vendas-372-em-2020.htm>

[2] SUTTO, Giovanna. Pandemia alavanca fabricação de robôs que têm 60 expressões faciais, 25 de jan. de 2021. Disponível em: <https://www.infomoney.com.br/carreira/profissionais-especialistas-devem-perder-emprego-para-robos-diz-professor-do-mit/>

[3] NAGWANI, Naresh Kumar; VERMA, Shrish. A comparative study of bug classification algorithms. International Journal of Software Engineering and Knowledge Engineering, v. 24, n. 01, p. 111-138, 2014.

[4] Basavanna. M , Dr. Shivakumar. M, 2019, An Overview of Path Planning and Obstacle Avoidance Algorithms in Mobile Robots, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 08, Issue 12 (December 2019).

[5] Ng, James, and Thomas Bräunl. "Performance comparison of bug navigation algorithms." *Journal of Intelligent and Robotic Systems* 50.1 (2007): 73-84.

[6] Zohaib, Muhammad, et al. "IBA: Intelligent Bug Algorithm–A novel strategy to navigate mobile robots autonomously." *International Multi Topic Conference*. Springer, Cham, 2013.

[7] McGuire, Kimberly N., Guido CHE de Croon, and Karl Tuyls. "A comparative study of bug algorithms for robot navigation." *Robotics and Autonomous Systems* 121 (2019): 103261.

[8] BINGHAM, Brain. Gazebo and Pioneer AT. NPS Wiki, 07 de jun. de 2016. Disponível em: <https://wiki.nps.edu/display/RC/Gazebo+and+Pioneer+AT>