Universidade Federal de Pernambuco
Centro de Informática
Sistemas de Informação - IF684
Relatório de Projeto

André De' Carli da Mota Silveira, Arthur Frade de Araújo,
Pedro Henrique Tôrres Santos, Sérgio Torres Teixeira Filho

1. Introdução

O projeto foi elaborado para a disciplina de Sistemas Inteligentes ministrada pelos professores Sérgio Queiroz e Cleber Zanchettin, como parte da avaliação da disciplina.

Com o propósito de aprofundar o conhecimento dos alunos nas áreas de Inteligência Artificial, Agentes Inteligentes e Resolução de Problemas, o projeto proposto faz uso do simulador MobileSim para a navegação de robôs em mapas e desafia os alunos a desenvolver uma solução que controle o robô e o guie para determinado ponto do mapa sem ter conhecimento dos obstáculos do mesmo. A linguagem de programação escolhida pelo grupo para o desenvolvimento foi C++.

2. Descrição do Problema

O problema apresentado no projeto consiste em guiar o robô p3dx-nolaser no simulador MobileSim em um mapa semelhante ao office.map de uma maneira que ele chegue o mais rápido possível em um determinado ponto do mapa (Figura 1).
O ambiente onde o robô irá atuar é caracterizado como:

- Parcialmente Observável
- Agente Mútiplo
- Estocástico
- Sequencial
- Dinâmico
- Desconhecido
- Contínuo

Espaço de Estados: Coordenadas do mapa não ocupadas por obstáculos.
Estado Inicial: Coordenada do mapa onde o robô começa sua busca.
Estado Final: Coordenada do mapa onde o robô deve chegar para completar seu objetivo.
Operadores: Se locomover de uma posição do mapa para outra, livre de obstáculos.
Teste de Término: O robô estar posicionado nas coordenadas do objetivo.

O robô não deve possuir conhecimento prévio do mapa, caracterizando o problema como uma busca on-line em um ambiente desconhecido. Para perceber os obstáculos ao seu redor, o robô dispõe de sonares localizados como a Figura 2.

Haverá também um fator de competição no projeto, onde o seu robô programado competirá com outro das outras equipes para ver quem chega primeiro no objetivo, ou quem chegou mais perto caso nenhum deles chegar no objetivo.

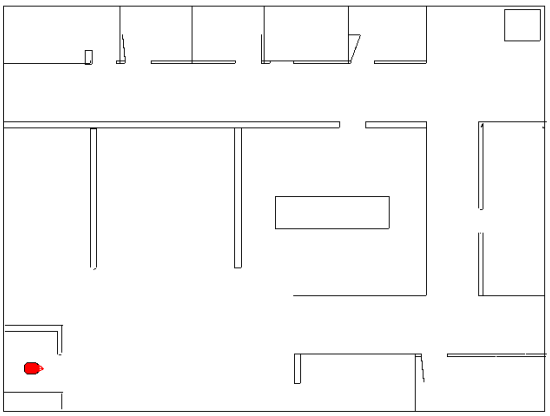

Figura 1. Mapa office.map com o robô p3dx-nolaser, em vermelho

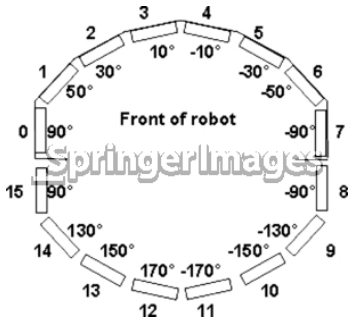

Figura 2. Disposição dos sonares no p3dx

3. Metodologia

3.1. Arquitetura do Agente

A arquitetura escolhida foi a de Agente Cognitivo Baseado em Objetivo. Esta arquitetura permite armazenar a evolução do ambiente (que é inicialmente desconhecido pelo agente) de acordo com a percepção de seus sensores, e escolher uma ação que vá (eventualmente) guiá-lo para o objetivo.

3.2. Descrição das Tentativas de Solucionar o Problema

Para representar os estados em que o agente se encontra, foi utilizado a posição (x,y) do mesmo. No entanto, como o espaço é contínuo e o limite de precisão dos sensores de localização do agente é finito, foi necessário discretizar o ambiente em um grid, onde cada estado possível de o agente estar é identificado pelo par (i, j), onde i e j são os índices da matriz que representa o ambiente.

A abordagem inicial foi tentar utilizar o algoritmo A*. No entanto, após discussão, ficou claro que o A* seria extremamente ineficiente para tal problema. Apesar do A*, ao final de sua execução, encontrar o melhor caminho entre dois pontos, seu método de busca levaria muito tempo para o p3dx. Isto se dá pois o agente teria que se deslocar por vários estados que não possuem saída para eliminar essa trajetória de sua solução.

Após descartar o uso de A*, encontramos o algoritmo D* Lite como proposto por [Koenig and Likhachev 2002]. O algoritmo se propõe a tratar com eficiência o caso

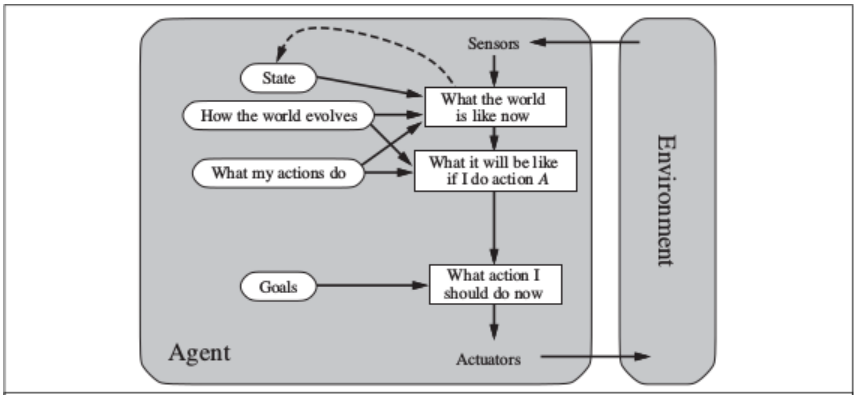

Figura 3. Estrutura do Agente Baseado em Objetivo. Retirado de [Russel and Norvig 2010]

da busca on-line num ambiente desconhecido. Traçando uma trajetória do objetivo até o agente e mudando a mesma o mínimo possível no caso de mudanças do ambiente (ao detectar um novo obstáculo, ou uma nova passagem onde antes havia um obstáculo), temos um ganho de eficiência se comparado ao A*. Isto porque não estamos necessariamente buscando o caminho mais curto, mas simplesmente chegar lá.

Na Figura 4 vemos um exemplo de funcionamento do D* Lite.

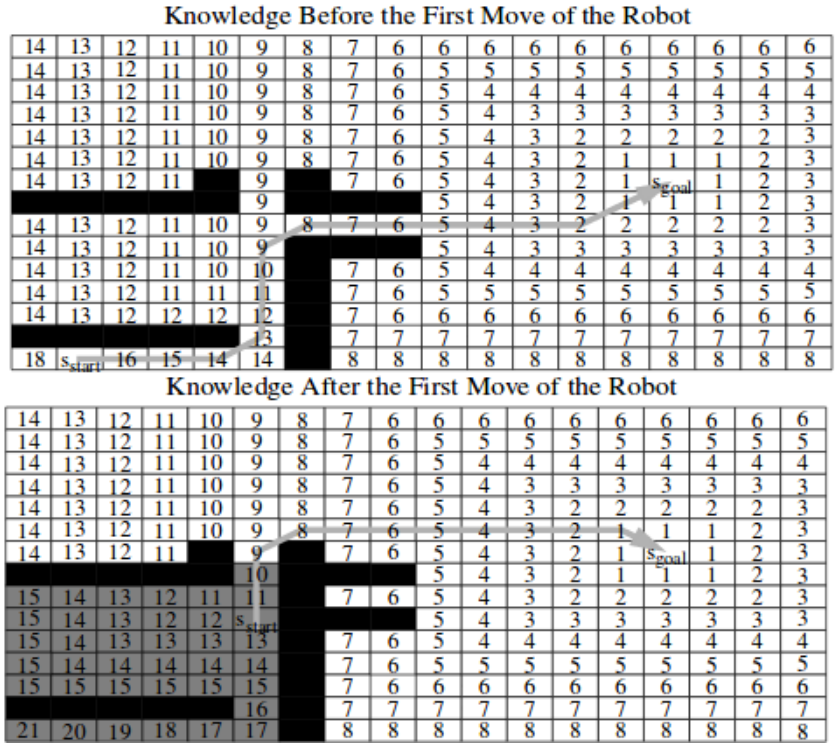

Figura 4. Exemplo de funcionamento do D* Lite.

O algoritmo D* Lite foi desenvolvido como uma mistura dos algoritmos já existentes LPA*(Lifelong Planning A*) [Koenig et al. 2004] e D*(Focused Dynamic A*)[Stenz 1994]. Ele usa funções do LPA* para repetidamente calcular o caminho mais

curto do vértice atual do robô para o vértice do objetivo à medida que os custos das arestas mudam enquanto o robô anda até o objetivo. Já do D*, ele usa a mesma estratégia de navegação, mas é algoriticamente diferente. O D* Lite não usa condições complexas como o D*, e simplifica a manutenção de prioridades com apenas uma condição de desempate das mesmas.

3.3. Descrição da Solução Implementada

Com a dificuldade de implementar os algoritmos pesquisados no problema proposto, tentamos criar nosso próprio método de fazer o robô desviar de obstáculos.

De acordo com a heurística utilizada, o robô prefere ir numa direção que minimize a distância de onde ele se encontra para o objetivo e que não haja obstáculos na trajetória. No caso de haver obstáculos, o agente procura iterativamente uma nova trajetória que desvie o mínimo possível da trajetória ótima e que não possua obstáculos. Para evitar que o robô volte a locais sem saída (dead ends), o algoritmo ”fecha”estes caminhos, colocando obstáculos ”virtuais”.

4. Análise dos Resultados Obtidos

Após exaustivas tentativas, não alcançamos implementar corretamente a lógica de fechar os caminhos sem saída. Com isso, o robô procura vários caminhos muito semelhantes aos já testados, levando muito tempo para conseguir sair desta condição.

5. Conclusões e Trabalhos Futuros

O problema de busca on-line em ambiente desconhecido se mostrou muito mais desafiador que o esperado. A implementação dos algoritmos pesquisados, que chegam no objetivo sem percorrer exaustivamente o espaço de estados, foi a maior dificuldade encontrada durante o desenvolvimento do projeto.

O desenvolvimento de um algoritmo para solucionar o problema tambem se mostrou muito trabalhoso, visto que foram necessários muitos testes para verificar o correto funcionamento do mesmo. Com uma baixa precisão dos sensores e dos diversos arredondamentos feitos para discretizar o espaço real, isto foi mais um obstáculo encontrado.

Após um estudo em cima do que foi feito no projeto, concluimos que, de fato, o algoritmo mais eficiente para solucionar este problema é o D* Lite. Com mais tempo para estudar o artigo encontrado [Koenig and Likhachev 2002], talvez seja possível implementá-lo no projeto proposto com bons resultados.

Referências

Koenig, S. and Likhachev, M. (2002). D* lite. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 476–483.

Koenig, S., Likhachev, M., and Furcy, D. (2004). Lifelong planning a*. In *Artificial Intelligence Journal*, volume 155(1-2), pages 93–146.

Russel, S. J. and Norvig, P. (2010). *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3rd edition.

Stenz, A. (1994). Optimal and efficient path planning for partially-known environments. In *Proceedings of the International Conference on Robotics and Automation*, pages 3310–3317.