Software Didático para Comparação entre Algoritmos de Busca em Espaço de Estados

José Eduardo Silva Gomes e Paulo Henrique Cruz Pereira

***Resumo*— O processo de ensino-aprendizagem quando realizado somente de forma expositiva não é tão eficiente e deve ser complementado com outras ferramentas didáticas para melhor compreensão e fixação dos conteúdos. O objetivo deste trabalho é implementar um programa computacional que simula a execução de algoritmos que realizam busca de caminho entre dois pontos, em um cenário virtual se desviando de obstáculos. Com este *software* didático é possível comparar o desempenho de três algoritmos de busca: Busca de Custo Uniforme (Dijkstra); Busca Gulosa pela Melhor Escolha e Busca A* (pronuncia-se A estrela).**

***Palavras-chave*— Ferramenta didática, inteligência artificial, algoritmos de busca.**

I. INTRODUÇÃO

Algoritmos que determinam o melhor trajeto entre dois pontos se desviando de obstáculos são conhecidos como algoritmos de busca de caminho ou *pathfinding*. Os algoritmos de busca podem ser avaliados sob quatro aspectos [5]:

- Completeza – Ele encontra a solução se ela existir?
- Otimização – Ele encontra a solução de menor custo?
- Tempo – Quanto tempo leva para encontrar a solução?
- Espaço – Quanto o algoritmo consome de memória para executar a busca?

Entre algoritmos de *pathfinding* a busca A* é a mais utilizada, devido ser completa, ótima e eficiente. No desenvolvimento de jogos, o algoritmo A* é provavelmente um dos mais, senão o mais utilizado algoritmo de *pathfinding* [2]. Ele também tem sido muito usado em sistemas robóticos autônomos e foi descrito pela primeira vez em 1968 por Peter Hart e Nils Nilsson [1].

A Busca de Custo Uniforme realiza uma busca ótima e completa, mas por ser uma busca cega (sem informação) utiliza uma estratégia baseada em tentativas de solução por força bruta, o que pode gerar um alto custo computacional [3]. Este algoritmo calcula o caminho de custo mínimo entre um nó de origem e um nó de destino. Para isso ele utiliza a função de avaliação de custo de caminho:

$$f(n) = g(n) \qquad (1)$$

Sendo $g(n)$ o custo do caminho já percorrido partindo da raiz até o nó n.

A Busca Gulosa pela Melhor Escolha é uma busca com informação ou busca heurística que procura minimizar os custos computacionais empregando, além da definição do problema, conhecimento específico a cerca do problema em questão. Ela não é ótima e nem completa, mas com uma boa heurística pode encontrar mais rapidamente uma solução, ocasionando um menor custo computacional [3]. Esta busca tenta expandir o nó mais próximo a meta, na suposição de que isso provavelmente levará a uma solução rápida [5]. Isto é feito baseado apenas na estimativa feita pela função heurística:

$$f(n) = h(n) \qquad (2)$$

Em que $h(n)$ é o custo estimado do caminho para se chegar ao objetivo partindo-se do nó n.

A busca A* combina estas duas estratégias, pois é ótima, completa e, com uma boa heurística, pode minimizar significativamente o custo computacional da busca, ou seja, a complexidade de tempo e de espaço [3]. O algoritmo de Busca A* utiliza como função de avaliação a soma das duas busca anteriores, ou seja:

$$f(n) = g(n) + h(n) \qquad (3)$$

Assim para a codificação de um software que compare estes três algoritmos foi implementado o algoritmo A* e depois com simples modificações foi possível simular os outros dois. O propósito da aplicação computacional desenvolvida é a execução dos três algoritmos em um mesmo ambiente, a fim de analisar o desempenho de cada estratégia de busca no que se refere à qualidade do caminho encontrado, o tempo e o consumo de recursos do sistema.

II. SOFTWARE DIDÁTICO

Quando a busca de caminho é realizada em um ambiente contínuo o número de nós gerados é muito amplo e isto pode tornar o custo computacional da busca elevado. Normalmente a área de busca é simplificada representando o ambiente através de uma grade. Este método particular reduz a área de procura a uma ordem simples bidimensional [4]. A figura 1 mostra a interface gráfica do *software*, nela pode-se observar que a área de busca foi discretizada em quadrados. Assim cada quadrado representa um nó na área de busca e o painel gráfico da área de busca pode ser associado a uma matriz bidimensional de 16x12 posições. Sendo que cada posição pode assumir os estados de livre (passável), obstáculo (não-passável), origem ou destino.

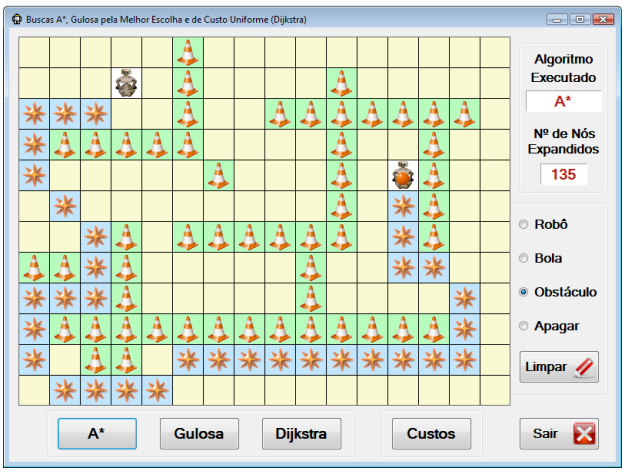

Fig. 1. Interface Gráfica.

José Eduardo Silva Gomes e Paulo Henrique Cruz Pereira, Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG, Varginha, MG, Brasil, E-mails: jeduardo@varginha.cefetmg.br, paulovga@varginha.cefetmg.br

Em um mesmo cenário foram analisados os caminhos encontrados por cada estratégia de busca. A figura 2 abaixo apresenta o caminho realizado pelo algoritmo A*.

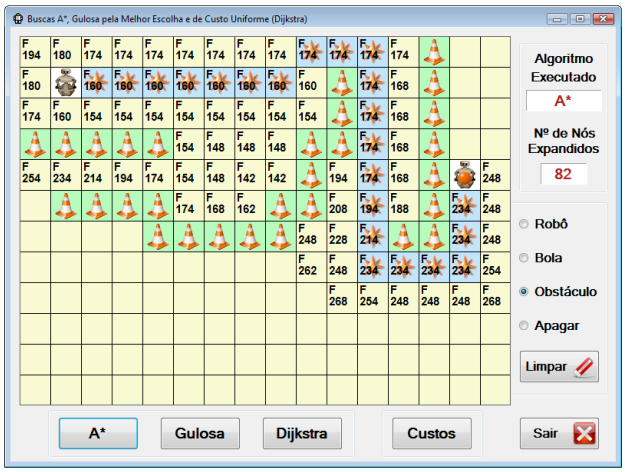

Fig. 2. Execução do algoritmo A*.

A figura 3 a seguir mostra o caminho resultante da execução do algoritmo de Busca Gulosa.

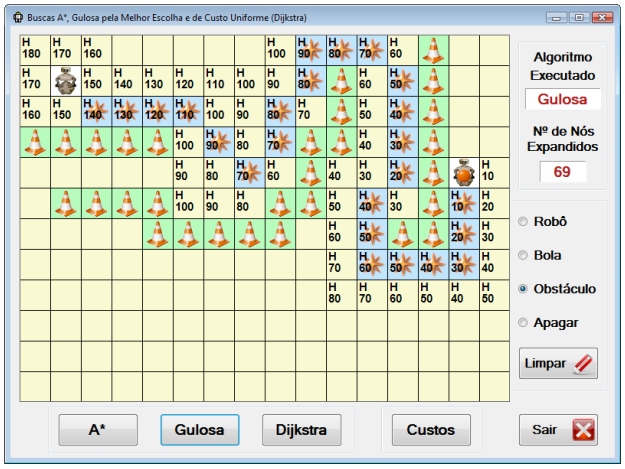

Fig. 3. Execução do algoritmo de Busca Gulosa.

A figura 4 abaixo mostra o caminho encontrado pelo algoritmo de Dijkstra (Custo Uniforme).

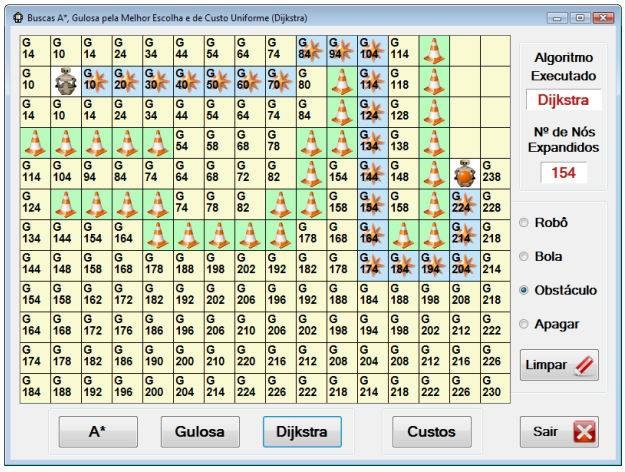

Fig. 4 – Execução do algoritmo de Dijkstra.

A busca A* produz uma solução ótima, portanto o caminho mostrado na figura 2 é o caminho mais curto entre a origem (robô) e o destino (bola). Analisando visualmente o caminho realizado na busca Gulosa pela Melhor Escolha, da figura 3, nota-se que inicialmente ela seguiu um caminho para baixo e depois mudou de direção e realizou praticamente o mesmo trajeto da busca A*. Confirmando, como visto anteriormente, que a busca Gulosa não é ótima. A figura 4 mostra que o algoritmo de Dijkstra (Busca de Custo Uniforme) encontrou o mesmo caminho obtido pelo algoritmo A*, pois o algoritmo de Dijkstra também realiza uma busca ótima.

A comparação anterior foi mensurada de forma empírica, já que foi baseada em percepção visual. No que tange à complexidade de espaço, ou seja, o consumo de memória do sistema, o *software* permite conclusões mais consistentes. No campo "N° de Nós Expandidos" das figuras 2, 3 e 4 pode-se notar que o algoritmo de Dijkstra expandiu um maior número de nós (154 nós) que os algoritmos de busca A* (82 nós) e de busca Gulosa (69 nós). No algoritmo de Dijkstra o valor da função de avaliação, *g(n)*, de cada nó expandido deve ser armazenado em memória, assim conclui-se que sua complexidade de espaço é maior que nos outros dois algoritmos. A diferença do número de nós expandidos nas buscas Gulosa e A* não é muito significativa, mas pode-se perceber nos ensaios realizados que a busca gulosa normalmente expande um menor número de nós. Estas inferências ratificam a afirmação, feita anteriormente, de que uma boa heurística, *h(n)*, pode minimizar a complexidade de espaço. Através do botão "Custos" o software também mostra, além dos caminhos encontrados, os cálculos realizados na busca do caminho mais curto até o objetivo (bola).

O programa de comparação possui uma área de busca reduzida e, consequentemente, um pequeno espaço de estados, que é o conjunto de todos os estados acessíveis a partir do estado inicial. Nesta situação os três algoritmos são bastante eficientes em relação à complexidade de tempo, ou seja, ao tempo gasto na busca do objetivo. Assim, neste *software*, a medida da velocidade de execução de cada algoritmo não retorna valores que permitem uma análise satisfatória em relação à complexidade de tempo.

III. Conclusão

Os resultados comparativos entre os algoritmos corroboraram com a fundamentação teórica, que aponta a estratégia A* como melhor em qualidade e em desempenho, quando comparada às buscas Gulosa e de Custo Uniforme (Dijkstra). Desta forma acredita-se que o sistema atingiu os objetivos propostos, atendendo aos requisitos levantados no início do trabalho, e que o *software* pode ser utilizado como ferramenta didática de auxílio ao processo de ensino de Algoritmos de Busca em Espaço de Estados na área de Inteligência Artificial.

Referências

[1] ALGORITMO A*. In: WIKIPÉDIA, a Enciclopédia Livre. Wikimedia, 2009. Disponível em: <http://pt.wikipedia.org/w/index.php?title=Algoritmo_A*&oldid=16058354>. Acesso em: 23 mai. 2016.

[2] D. M. Bourg, G. Seemann. AI for Game Developers. Cambridge: O'Reilly Media, 2004.

[3] J. E. S. Gomes; P. H. C. Pereira. "Técnicas de Busca de Caminhos: uma Revisão Bibliográfica". ENCOM - Encontro Anual do IECOM em Comunicação, Redes e Criptografia. UFCG, Campina Grande-PB, 2015.

[4] P. Lester. A* Pathfinding for Beginners. Policyalmanac, 2005. Disponível em: <http://www.policyalmanac.org/games/aStarTutorial.htm>. Acesso em: 17 jan. 2016.

[5] S. Russell, P. Norvig. Artificial Intelligence: A Modern Approach. 3th Ed. New Jersey: Pearson, 2010.