Planejamento de trajetória em ambientes com prioridades dinâmicas

Heitor L. Polidoro
Universidade de São Paulo
ICMC - Instituto de Ciências Matemáticas e de Computação
São Carlos - SP, Brasil
heitor.polidoro@gmail.com

Denis F. Wolf
Universidade de São Paulo
ICMC - Instituto de Ciências Matemáticas e de Computação
São Carlos - SP, Brasil
denis@icmc.usp.br

***Resumo*—A robótica móvel é uma área de pesquisa que está obtendo grande atenção da comunidade científica. O desenvolvimento de robôs móveis autônomos, que sejam capazes de interagir com o ambiente, aprender e tomar decisões corretas para que suas tarefas sejam executadas com êxito é o maior desafio em robótica móvel. O desenvolvimento destes sistemas inteligentes e autônomos consiste em uma área de pesquisa multidisciplinar considerada recente e extremamente promissora que envolve; por exemplo, inteligência artificial, aprendizado de máquina, estimação estatística e sistemas embarcados. Dentro desse contexto, esse trabalho aborda o problema de navegação e monitoramento de ambientes utilizando robôs móveis. Dada uma representação do ambiente (mapa topológico) e uma lista com urgências de cada uma das regiões do mapa, o robô deve estimar qual o percurso mais eficiente para monitorar esse ambiente. Uma vez que a urgência de cada região não visitada aumenta com o tempo, o trajeto do robô deve se adaptar a essas alterações. Entre as diversas aplicações práticas desse tipo de algoritmo, destaca-se o desenvolvimento de sistemas de segurança móveis inteligentes.**

***Keywords*-Robótica Móvel; Prioridades dinâmicas;**

I. Introdução

A robótica é uma tecnologia utilizada para auxiliar ou substituir o homem em tarefas que o ser humano não é capaz de executar; em tarefas que podem ser automatizadas; em tarefas em ambientes com alto grau de periculosidade, como fundo do mar, incêndios, desarmamento de bombas, áreas com contaminação radioativa ou com gases tóxicos e em tarefas repetitivas como uma linha de produção industrial, por exemplo.

A robótica consiste em uma área multidisciplinar de pesquisa, envolvendo desde elementos de engenharia mecânica, elétrica, computação até áreas de humanas como psicologia e estudos comportamentais. O que diferencia a robótica móvel de outras áreas de pesquisa em robótica, é a sua ênfase nos problemas relacionados com locomoção em ambientes complexos, que se modificam dinamicamente, compostos tanto por obstáculos fixos quanto por obstáculos móveis. Para operar nesses ambientes o robô deve ser capaz de adquirir e utilizar conhecimento sobre o ambiente, tais como: estimar posições dentro do ambiente (sua posição, de um obstáculo, de um *landmark*, de uma meta), reconhecer obstáculos, e responder em tempo real às situações que podem ocorrer nesses ambientes. As tarefas de perceber o ambiente, localizar-se no ambiente e mover pelo ambiente são problemas fundamentais da robótica móvel [1].

A motivação deste projeto é desenvolver um sistema de monitoramento de ambientes internos. Existem diversas aplicações práticas para esse tipo de aplicação. Dentre elas, pode-se citar o desenvolvimento de um robô vigia que monitora um ambiente, dando ênfase para áreas

de maior importância; um robô que monitora a temperatura e umidade de ambientes onde esse fator é crítico; um robô que faz coleta de lixo ou correspondência; um robô para fazer limpeza de ambientes; um robô para fazer entregas de remédios a pacientes em um hospital e etc.

II. Desenvolvimento e Resultados

Para determinar a trajetória do robô foram criadas duas soluções, uma *Offline* e outra em Tempo Real, essas soluções serão confrontadas em dois critérios de avaliação, um baseado no Grau de Urgência Total e o outro baseado na freqüência relativa de cada sala.

A. Critérios de Avaliação

Os algoritmos e estratégias serão então testados e avaliados. Serão utilizados dois critérios de comparação, um critério comparando a freqüência relativa de cada sala com sua prioridade relativa, onde o melhor resultado é aquele em que a freqüência relativa se aproximar mais da prioridade relativa. Esse cálculo é feito somando as diferenças quadráticas entre a prioridade relativa e a freqüência relativa de cada sala ($\sum_{i=0}^{n}(P_i/P_t - F_i/F_t)^2$). Isto indica que o algoritmo ou estratégia se manteve fiel à definição do problema: *Áreas de maior importância devem ser visitadas mais frequentemente.*

O segundo critério é um gráfico mostrando a progressão da somatória dos graus de urgência de todas as salas (Grau de Urgência Total $\sum_{i=0}^{n} U_i$) , no qual o melhor resultado consiste em manter o menor valor da somatória dos graus de urgência; mostrando que o algoritmo ou estratégia levou o robô às salas com maior eficiência. O grau de urgência U é calculado multiplicando-se a prioridade relatica da sala P/P_t pelo tempo decorrido desde a última visita t ($U_i = P_i/P_t \times t_i$). A casa visita o t é zerado.

B. Metodologia

Para a solução do problema descrito, o robô tem uma descrição completa do ambiente em que atua (mapa). Foram utilizados somente dois sensores: Odômetro para localizar o robô no mapa e o LASER para o desvio de obstáculos. O controle do robô foi desenvolvido utilizando a biblioteca Player/Stage [2]. Para desvio de obstáculos utilizou-se a técnica VFH que já vem implementada no Player e se mostrou suficiente para o projeto.

Cada ambiente tem um mapa topológico [3]. O robô deve utilizar esse mapa para se locomover de uma sala para outra no ambiente. Para os algoritmos e estratégias determinarem a seqüência de salas a serem visitadas, é considerado um grafo completo com todas as salas, pois para determinar o melhor caminho entre uma sala e outra será utilizado o algoritmo de Dijkstra [4] no mapa topológico.

Como um dos critérios de avaliação está relacionado ao tempo que o robô fica sem visitar as salas é possível deduzir que a solução seja um ciclo.

Considerando: um ambiente com S salas; um ciclo hamiltoniano C qualquer [5]; C_i é a i-ésima sala visitada no ciclo; uma velocidade constante do robô (tanto linear quanto angular) e Δt_i o tempo para sair da sala i e chegar na sala $i+1$. Como o tempo de viagem entre as mesmas salas é constante todos os Δt_i são constantes, portanto o tempo total do ciclo T é constante ($\sum_{i=1}^{S} \Delta t_i = cte$). Se o tempo do ciclo é constante, o tempo que o robô demora para revisitar cada sala é constante igual a T. Mas isso não é interessante para solução do problema pois o grau de urgência de cada sala é diferente, então se uma sala tem uma prioridade muito alta, seu grau de urgência vai ser muito alto até o robô revisitá-la.

Para resolver esse problema, basta fazer com que o robô revisite essa(s) sala(s) mais de uma vez no ciclo. Então supondo um ciclo de tamanho n sendo $n \geq S$ o tempo total do ciclo continua constante ($\sum_{i=1}^{n} \Delta t_i = cte$). Portanto a solução do problema consiste em encontrar esse ciclo.

C. Soluções

Foram definidos dois tipos de soluções: uma solução ***offline***, ou seja, a solução é calculada em um computador e depois informada ao robô

qual seqüência de sala ele deve seguir, essa seqüência não é alterada. E a outra é uma solução **tempo real** em que o robô define para qual sala deve ir durante a execução do algoritmo, baseando suas decisões no que está acontecendo no momento.

1) Offline: Esse método consiste em explorar as diversas possíveis combinações de seqüências de salas para encontrar a seqüência ótima antes de informar ao robô qual seqüência de salas deve seguir. Um programa chamado `gerador` gera as possíveis seqüências de salas, essas seqüências são fornecidas ao programa `avaliador` que, baseado no critério de maior Grau de Urgência Total, analisa a seqüência e retorna uma avaliação ao `gerador` que tomará a decisão de: descartar a seqüência, preservar a seqüência para gerar futuras seqüências ou guardar a seqüência como possível seqüência ótima.

Gerador: O programa `gerador` utiliza uma classe (tipo de variável) chamada *Agente* contendo 4 atributos: `vertice`, vértice no qual o agente se encontra no momento; `caminho`, vetor de salas que guarda a seqüência de salas que o agente percorreu até chegar no vértice atual; `avaliacao`, avaliação da seqüência de salas do agente e `tempo`, tempo em segundos que o robô levou para percorrer a seqüência de salas até o vértice atual.

O programa inicia criando um *Agente* em cada sala do mapa para explorar as possibilidades do robô começar em cada sala, esses *agentes* são organizados em uma fila de candidatos. O laço principal do programa consiste em retirar um *agente* da fila e para cada sala i é criado um novo *agente*, simulando que o *agente* retirado da fila navegou até a sala i. Em cada novo *agente* é atualizada a seqüência de salas visitadas e o agente é reavaliado. Se a avaliação for um valor negativo o *agente* é inserido na fila de *agentes*. Se o valor retornado for 0 (zero) o *agente* é descartado. Caso o avaliador retorne um valor positivo e esse valor é menor que a avaliação da atual seqüência ótima, o novo *agente* contendo a seqüência melhor é guardado como possível seqüência ótima. Como o algoritmo somente insere na fila os *agentes* que podem gerar seqüências ótimas, o critério de parada é quando não existe mais *agentes* na fila, o programa então retorna o *agente* com a seqüência ótima.

Avaliador: O programa chamado `avaliador` é utilizado para avaliar os candidatos a seqüência de salas ótima do mapa analisado gerado pelo programa `gerador` e retornando ao mesmo um valor numérico.

O `avaliador` inicia a simulação do robô navegando pelo mapa. Durante a simulação, a cada visita de sala o `avaliador` mede o Grau de Urgência Total e salva o maior valor, esse valor vai ser a avaliação do caminho neste mapa. Após a simulação o `avaliador` verifica se o caminho não visita todas as salas o `avaliador` retorna a avaliação com valor negativo, se todas as salas foram visitadas o `avaliador` verifica se o caminho é um *loop*, ou seja, se o caminho começa e termina no mesmo vértice, caso contrário ele também retorna a avaliação em valor negativo. Se o caminho for um *loop* e visita todas as salas o `avaliador` retorna o valor da avaliação.

Para critério de parada o `avaliador` compara o estado das salas, ou seja, se para cada sala, o Grau de Urgência e o número de visitas é o mesmo ao término do caminho.

2) Tempo Real: A solução em tempo real inicia as prioridades de todas as salas em zero, cada sala recebe um valor correspondente à chance dessa sala gerar uma emergência. Emergência é quando a sala solicita que o robô vá visitá-la, como uma lixeira cheia ou a sala não ser visitada por um tempo mínimo determinado por exemplo. A cada emergência gerada a prioridade da sala é acrescida de uma unidade. O robô segue o paradigma de ir à sala de maior grau de urgência, visita a sala e procura a próxima sala de maior grau de urgência.

A chance de emergência que cada sala recebe é proporcional às prioridades, para fazer o cálculo basta dividir a prioridade da sala pela soma das prioridades do mapa.

$$p_s = \frac{P_s}{\sum_{i=1}^{TotalSalas} P_i} \quad (1)$$

Isso faz com que a soma das chances seja igual a um.

D. Resultados

Para a solução em Tempo Real, por se tratar de uma solução baseada em números aleatórios, foram executados 100 repetições em cada mapa.

Para os testes foram criados sete mapas, cada qual com uma peculiaridade diferente como mostra a tabela I

Tabela I
MAPAS PARA TESTES

Nome	N# de Salas	Prioridades	Formato
A	4	Iguais	Quadrado com ligação entre todas as salas.
B	4	Diferentes	Idem ao mapa B.
C	5	Diferentes	Cinco salas em formato um X com ligações entre as salas das pontas.
D	5	Diferentes	Cinco salas em formato de X com somente a sala do meio ligando as outras.
E	6	Iguais	Seis salas em formato de espinha de peixe.
F	6	Diferentes	Idem ao Mapa F.
G	8	Diferentes	Oito salas simulando um apartamento real.

A Tabela II mostra as seqüência de salas geradas pelo programa `gerador`.

Tabela II
SEQÜÊNCIAS DE SALAS GERADAS PELO `GERADOR`

Mapa	Seqüência de Salas
A	1 2 4 3
B	1 3 4 2 4 3
C	1 2 4 5 3 2 4 5 3
D	1 2 3 4 5
E	1 5 2 6 3 7
F	1 2 6 5 1 7 3 5
G	1 12 9 11 5 7 3 4

As prioridades relativas finais da solução em Tempo Real foram próximas às prioridades relativas iniciais do mapa, em todos os mapas, como mostra a Tabela III.

A Tabela III mostrou também que as freqüências relativas dos todos os mapas, tanto da solução *Offline* quanto a em Tempo Real, foram próximas às prioridades relativas iniciais do mapa excetuando-se: no mapa B as freqüências relativas das salas 1 e 2 da solução *Offline* são aproximadamente o dobro das suas respectivas prioridades relativas iniciais do mapa, porém, em números absolutos, é uma diferença de aproximadamente 0,08; no mapa C a sala 1 da solução *Offline* foi maior que o dobro da sua prioridade relativa inicial do mapa, entretanto, a diferença numérica absoluta é de aproximadamente 0,06; nos mapas D e G as freqüências relativas da solução *Offline* foram aproximadamente todas iguais, isse se deve ao fato de que a seqüência de salas a serem visitadas gerada pelo `gerador` manda o robô visitar cada sala apenas uma única vez dentro do *loop* e no mapa F as freqüências relativas da solução *Offline* foram diferentes das prioridades relativas iniciais do mapa, visitando as salas 1 e 5 o dobro de vezes que as demais salas, pois na seqüência determinada pelo `gerador` as salas 1 e 5 aparecem duas vezes cada uma.

Por fim, a Tabela III mostrou também que nos mapas A e E a solução *Offline* foi melhor que a solução em Tempo Real através da diferença quadrática e nos mapas B, C, D, F e G a solução em Tempo Real foi melhor que a solução *Offline* através da diferença quadrática.

Todos os gráficos tiveram o mesmo comportamento, a solução *Offline* mantendo os valores dentro de uma faixa e a solução em Tempo Real subindo e estabilizando-se em uma faixa com valores superiores à solução *Offline* (Figura 1).

1) Análise Comparativa: Foi calculada uma média do Grau de Urgência Total de cada solução em cada mapa com os últimos dez mil segundos (Tabela IV); analisando essa média nota-se que existe uma tendência de que quanto maior o número de salas mais eficiente será a solução *Offline* em relação à solução em Tempo Real, como mostra o gráfico na Figura 2.

Tabela III
RESULTADOS

			Prioridades Relativas		Freqüências Relativas	
Mapa	Sala	Prioridade	Offline	Tempo Real	Offline	Tempo Real
A	1	1	0,25000	0,24989	0,25021	0,25114
	2	1	0,25000	0,24860	0,24993	0,25263
	3	1	0,25000	0,25224	0,24993	0,24877
	4	1	0,25000	0,24927	0,24993	0,24746
	Diferença Quadrática				0,00000	0,00002
B	1	1	0,08333	0,08470	0,16670	0,10130
	2	1	0,08333	0,08331	0,16670	0,09994
	3	5	0,41666	0,41141	0,33323	0,40206
	4	5	0,41666	0,42058	0,33323	0,39670
	Diferença Quadrática				0,02782	0,00121
C	1	1	0,04762	0,04983	0,11134	0,05670
	2	5	0,23810	0,24044	0,22234	0,23729
	3	5	0,23810	0,23870	0,22200	0,23807
	4	5	0,23810	0,23501	0,22234	0,23210
	5	5	0,23810	0,23602	0,22200	0,23583
	Diferença Quadrática				0,00507	0,00012
D	1	1	0,04762	0,04598	0,20009	0,05481
	2	5	0,23810	0,23794	0,20009	0,23908
	3	5	0,23810	0,24530	0,20009	0,23633
	4	5	0,23810	0,23984	0,20009	0,23275
	5	5	0,23810	0,23094	0,19965	0,23703
	Diferença Quadrática				0,02906	0,00009
E	1	2	0,16667	0,16990	0,16694	0,16869
	2	2	0,16667	0,16613	0,16694	0,16483
	3	2	0,16667	0,16571	0,16640	0,16692
	5	2	0,16667	0,16339	0,16694	0,16832
	6	2	0,16667	0,16720	0,16640	0,16179
	7	2	0,16667	0,16768	0,16640	0,16945
	Diferença Quadrática				0,00000	0,00004
F	1	5	0,41667	0,41297	0,25000	0,35364
	2	1	0,08333	0,09396	0,12500	0,10120
	3	2	0,16667	0,16624	0,12500	0,17144
	5	2	0,16667	0,17418	0,25000	0,17798
	6	1	0,08333	0,07849	0,12500	0,09488
	7	1	0,08333	0,08427	0,12500	0,10087
	Diferença Quadrática				0,04167	0,00488
G	1	5	0,15152	0,15426	0,12525	0,14970
	3	5	0,15152	0,15233	0,12475	0,15172
	4	3	0,09091	0,08996	0,12475	0,09717
	5	3	0,09091	0,08924	0,12475	0,09447
	7	5	0,15152	0,14934	0,12475	0,14197
	9	4	0,12121	0,12566	0,12525	0,12442
	11	4	0,12121	0,12133	0,12525	0,12004
	12	4	0,12121	0,11789	0,12525	0,12050
	Diferença Quadrática				0,00446	0,00016

III. CONCLUSÃO

Segundo o critério de avaliação de freqüência relativa, a solução em Tempo Real foi superior à solução *Offline*, pois foi mais eficiente em cinco dos sete testes e na soma total das diferenças quadráticas (0,10812 da solução *Offline* contra 0,00652 da solução em Tempo Real) e segundo o critério de avaliação de Grau de Urgência Total a solução *Offline* foi melhor que a solução em Tempo Real em todos os testes realizados e

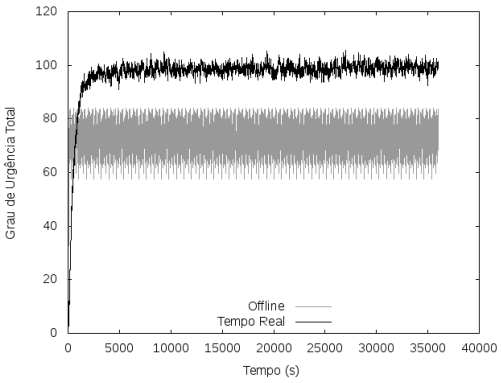

Figura 1. Gráfico do Mapa G

Tabela IV
Média dos Últimos Dez Mil Segundos.

Mapa (N# de Salas)	*Offline*	Tempo Real
A (4)	25,000	28,150
B (4)	20,980	28,346
C (5)	28,905	38,753
D (5)	39,719	50,326
E (6)	58,527	72,622
F (6)	55,557	66,711
G (8)	73,318	100,217

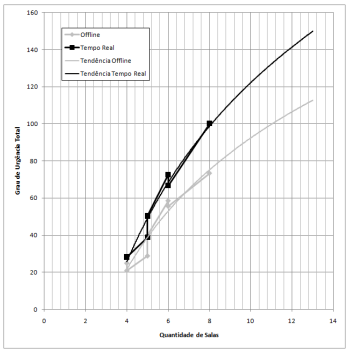

Figura 2. Tendência de Desempenho

o gráfico da Figura 2 mostrou que existe um tendência de que quanto maior o número de salas maior será a eficiência da solução *Offline* em relação à solução em Tempo Real.

Era esperado que a solução *Offline* tivesse um desempenho pior que a solução em Tempo Real no critério de avaliação de freqüência relativa, pois a avaliação das seqüências de salas à visitar é baseada somente no Grau de Urgência Total; e o desempenho das soluções segundo o critério de avaliação de Grau de Urgência Total mostra que o robô, seguindo uma heurística simples, tem um desempenho inferior do que uma solução que determina a seqüência de salas a serem visitadas, baseada no Grau de Urgência Total, antes de colocar o robô no ambiente.

Por fim, o que determina qual solução deve ser utilizada depende da aplicação. Se para a aplicação a freqüência relativa for mais importante do que manter o Grau de Urgência baixo, deve-se escolher a solução em Tempo Real, porém se para a aplicação for mais importante manter o Grau de Urgência baixo deve-se escolher a solução *Offline*.

Para dar continuidade ao projeto o próximo é explorar a utilização de mais de um robô para realizar a tarefa, cada robô monitorar um conjunto de salas divididas por prioridade ou proximidade, ou todos os robôs podem monitorar todas as salas etc.

Referências

[1] F. J. Heinen and F. Osório, "Sistema de controle híbrido para robôs móveis autônomos," *WTDIA/SBIA - Workshop De Teses E Dissertações D Inteligência Artificial, 2002*, 2002.

[2] Player, "Player project," http://playerstage.sourceforge.net/, 2010. [Online]. Available: http://playerstage.sourceforge.net/

[3] R. Siegwart and I. R. Nourbakhsh, *Introduction to Autonomous Mobile Robots*. MIT Press, 2004.

[4] G. Dudek and M. Jenkin, *Computational Principles of Mobile Robotics*. Cambridge University Press, 2000.

[5] M. N. Arenales, V. Armentano, R. Morabito, and H. Yanasse, *Pesquisa operacional*. Elsevier Campus, 2007.