Planejamento de Rotas de Robôs Móveis: Estudo da Viabilidade de Uma Abordagem Baseada em Algoritmos Genéticos em um Ambiente Multiagente

Tauã M. Cabreira[1], Marilton S. de Aguiar[1 3], Graçaliz P. Dimuro[1,2]

[1]Programa de Pós-Graduação em Modelagem Computacional
[2]Programa de Pós-Graduação em Computação
Universidade Federal do Rio Grande (FURG)
96.203-900 - Rio Grande - RS - Brasil

[3]Programa de Pós-Graduação em Computação
Universidade Federal de Pelotas (UFPel)
96.010-610 - Pelotas - RS - Brasil

tsiad.taua@gmail.com, marilton@inf.ufpel.edu.br, gracaliz@gmail.com

Abstract. *This paper describes an approach of genetic algorithms for the path planning of mobile robots in static and dynamic environments. With the software Netlogo, used in simulations of multi-agent applications, we developed a seminal model for the given problem. The model, which contains a robot and scenaries with or without obstacles, is responsible for determining the best path used by a robot to achieve the objective state in a shorter number of steps, avoiding collisions. Additionally, a performance evaluation of this model in comparison with A* algorithm is presented.*

Resumo. *Este artigo descreve uma abordagem de algoritmos genéticos para o planejamento de rotas de robôs móveis em ambientes estáticos e dinâmicos. Através do software Netlogo, usado em simulações de ambientes multiagentes, foi desenvolvido um modelo seminal para o problema em questão. O modelo, composto por um robô e cenários sem ou com obstáculos fixos e móveis, é responsável por determinar a melhor rota para alcançar o estado objetivo no menor número de passos, evitando colisões com os obstáculos. Além disso, uma avaliação de desempenho deste modelo em comparação ao algoritmo A* é apresentada.*

1. Introdução

O problema de planejamento de rotas de robôs móveis consiste em determinar uma rota para um robô, em um ambiente estático e/ou dinâmico, capaz de levá-lo do estado inicial ao estado objetivo através do melhor caminho, ou seja, o caminho mais curto, evitando possíveis colisões com obstáculos. Este problema já possui diversas abordagens sugeridas por diferentes pesquisadores, dentre as quais se pode citar a *Dead-reckoning*. No entanto, este e outros métodos apresentam ineficiências motivadas pelo acúmulo de erros (TU, 2003).

A abordagem de algoritmos genéticos surgiu como uma importante alternativa em diversos trabalhos recentes relacionados ao problema do planejamento de rotas de

robôs. Além da extensa possibilidade de variações, configurações e modificações, a abordagem de algoritmos genéticos possibilita ainda a agregação de outras técnicas que complementem e/ou aprimorem os resultados obtidos. Um Algoritmo Genético (AG) é uma variante de busca em feixe estocástica, na qual os estados sucessores são gerados pela combinação de dois estados pais, em vez de serem gerados pela modificação de um único estado (RUSSEL e NORVIG, 2003).

Os AG's foram inventados por John Holland nos anos 60, cujo principal objetivo não foi desenvolver algoritmos para solucionar problemas específicos, mas dedicar-se ao estudo formal do fenômeno de evolução, como ocorre na natureza, e desenvolver maneiras de importá-lo aos sistemas de computação (AGUIAR, 1998). Neste modelo, tem-se inicialmente um conjunto de estados (indivíduos) gerados aleatoriamente intitulado de população. Cada indivíduo é representado por um único cromossomo, contendo a codificação (genótipo) de um candidato à solução de um determinado problema (CONCILIO, 2000).

Em (SADATI e TAHERI, 2002), a abordagem de algoritmos genéticos foi combinada com o uso das Redes Neurais de Hopfield para evitar a colisão com obstáculos no processo de *crossover* entre dois caminhos distintos. Através do uso deste tipo de rede neural, existe a possibilidade de adicionar novos nodos na rota do robô, gerando desvios ao longo do trajeto e, evitando assim, a colisão com obstáculos. Já em (LEI, WANG e WU, 2006), o campo de potencial numérico foi agregado aos algoritmos genéticos, sendo levado em consideração o *feedback* da posição e dos movimentos dos obstáculos para evitar colisões através de replanejamento da rota.

Além da agregação de novas técnicas ao uso de algoritmos genéticos na resolução do problema de planejamento de rotas, também são realizadas mudanças nas características tradicionais do modelo dos AG's. Este é o caso do trabalho de (LI, TONG, XIE e ZHANG, 2006), onde um algoritmo genético híbrido é proposto com a finalidade de evitar uma convergência prematura no modelo. Para tal, uma estratégia de controle *fuzzy* é empregada para o ajuste das probabilidades de *crossover* e mutação, tornando o modelo autoadaptativo.

Conhecimentos adicionais sobre o problema em questão também podem ser empregados no modelo de algoritmos genéticos. No trabalho de (HU e YANG, 2004), o conhecimento do domínio foi incorporado em operadores especializados, combinados com técnicas de busca local. Neste modelo, foram desenvolvidos alguns operadores dedicados especialmente ao refinamento das soluções, onde se podem ajustar linhas entre dois pontos que atravessam um obstáculo, um ponto que esteja localizado justamente sobre um obstáculo ou até mesmo o posicionamento dos nodos.

Através dos trabalhos pesquisados e analisados, percebe-se o uso das mais variadas técnicas em conjunto com os algoritmos genéticos. A própria abordagem dos algoritmos genéticos muda drasticamente conforme a proposta de trabalho desenvolvida, variando na modelagem do ambiente, operadores utilizados, composição dos cromossomos, etc. A proposta deste trabalho é fazer uso de uma abordagem de algoritmos genéticos para o problema do planejamento de rotas de robôs móveis utilizando um ambiente de desenvolvimento multiagente.

Este trabalho está organizado como descrito a seguir. Na Seção 2, apresenta-se o modelo de planejamento de rotas de robôs, bem como a descrição detalhada da interface e das funcionalidades do mesmo. Na Seção 3, descrevem-se os testes de desempenho comparativos realizados entre o A* e o AG, além dos resultados obtidos através das simulações. Por fim, a Seção 4 conclui o trabalho e sinaliza a realização de trabalhos futuros.

2. O Modelo de Planejamento de Rotas de Robôs

A abordagem de algoritmos genéticos é uma técnica robusta e flexível, com eficiência comprovada na resolução de diversos tipos de problemas. Através do uso de AG, pretende-se desenvolver simulações em diversos tipos de ambientes estáticos e dinâmicos, visando analisar a capacidade de evitar colisões e de determinar a escolha da melhor rota.

A plataforma de desenvolvimento adotada foi o software *Netlogo*, um ambiente de desenvolvimento de modelos multiagentes, baseado na linguagem Logo, amplamente utilizado por estudantes, professores e pesquisadores (WILENSKY, 2011). Neste caso, fez-se uso dos agentes para a representação do robô e dos obstáculos, visto que o software possibilita, entre outras opções, a movimentação e interação entre os agentes, bem como a modelagem de um AG.

O modelo é composto por um robô, alguns obstáculos fixos e obstáculos móveis – a quantidade de obstáculos móveis pode ser alterada através da interface do aplicativo. Inicialmente, o robô projeta a melhor rota, baseando-se em um ambiente estático. Ao concluir o processo, o robô começa a realizar a sua rota, ao mesmo tempo em que os obstáculos começam a se movimentar, caracterizando um ambiente dinâmico. Ao encontrar um obstáculo pelo caminho, o robô recalcula a rota, redirecionando seu caminho para evitar as colisões. Ao atingir o estado objetivo, o robô para e o procedimento conclui-se.

2.1. O Fluxo da Simulação

Inicialmente, ao configurar o ambiente, é gerada uma população inicial aleatória de indivíduos, onde cada um destes indivíduos é composto por um *cromossomo* que constitui uma determinada rota. Todos os indivíduos que compõem a população são possíveis candidatos à solução do problema proposto.

Neste instante, no entanto, encontram-se no modelo apenas soluções geradas aleatoriamente sem nenhum critério de escolha. Encontram-se ainda no modelo obstáculos estáticos e móveis que poderão interferir na rota determinada para o robô. Este é o cenário inicial configurado a partir do botão SETUP da interface. Ao iniciar a simulação, através do botão GO, é dado início aos processos que constituem um algoritmo genético, que basicamente são a reprodução e a mutação, além da especificação da função de avaliação de cada novo indivíduo gerado. Os processos que constituem o algoritmo genético são executados por um determinado número de vezes, intitulado de gerações ou épocas. Ao término da execução destes procedimentos, o melhor indivíduo, ou seja, a melhor solução para o problema é retornada.

Em ambiente estático, onde os obstáculos estão imóveis, estes procedimentos seriam suficientes. Porém, o problema consiste em um ambiente estático e dinâmico, onde os obstáculos podem se movimentar livremente pelo ambiente, afetando diretamente a rota do robô. Desta forma, foram implementados procedimentos de detecção de obstáculos para evitar colisões do robô com os obstáculos. Portanto, a determinação da melhor rota para o robô, com ponto de partida e de chegada definidos, em um ambiente estático e dinâmico, consiste na busca pela rota mais breve e sem colisões, contendo ainda, a capacidade de identificação de obstáculos presentes no caminho e de recálculo da rota para alcançar o estado objetivo.

2.2. A Caracterização da Ferramenta

O ambiente de desenvolvimento do Netlogo é composto por três abas principais: *interface*, *information* e *procedures*. Na aba *interface*, pode-se controlar, executar e visualizar todas as ações do modelo. Por sua vez, na aba *information*, são acrescentadas todas as informações e características do modelo, bem como a explicação de seu funcionamento. E por fim, na aba *procedures*, têm-se todas as funções do modelo programadas na linguagem Logo. Nas seções que seguem, serão descritos os elementos que compõem a *interface* e as funções relacionadas a estes elementos implementadas na aba *procedures*. A explicação adotada na aba *information* dar-se-á ao longo do texto.

2.3. A Interface

Na aba *interface* do Netlogo, adicionaram-se diversos elementos e botões para a configuração, a execução e o controle da simulação do problema de planejamento de rota para robôs móveis, conforme apresenta a Figura 1.

Os botões SETUP e GO são os responsáveis pela configuração e execução do modelo. Inicialmente, o ambiente é composto por uma população (invisível) de candidatos à melhor rota e obstáculos fixos (cor laranja) e móveis (cor azul) distribuídos pelo mapa. Ao iniciar a simulação através do botão GO, o algoritmo genético é executado por um determinado número de gerações, que é definido pelo próprio usuário através do *slider* (botão que corre horizontalmente) GERACOES.

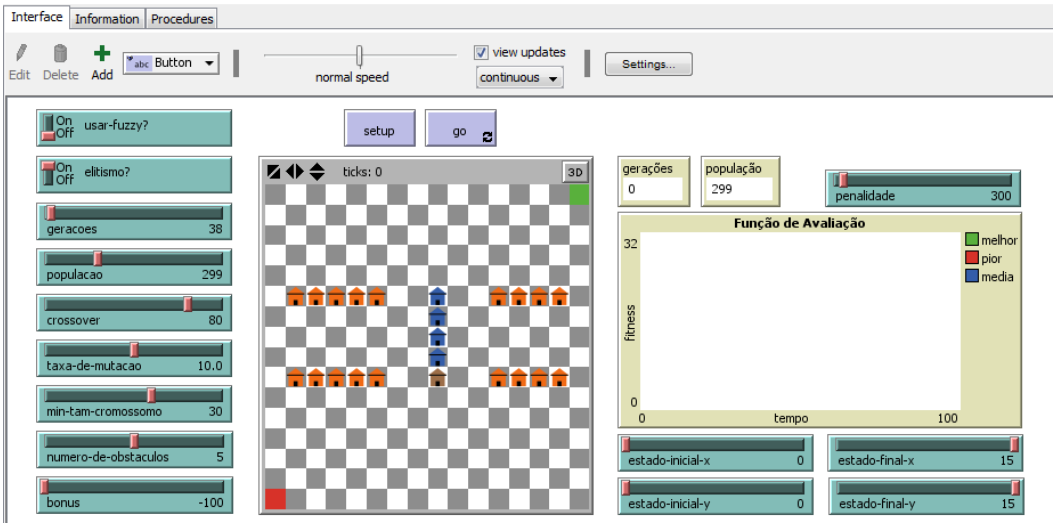

Figura 1. Interface da Ferramenta e Visão Geral do Netlogo

Além dos botões SETUP e GO, existem outros *sliders* cuja função é configurar os parâmetros adotados no algoritmo genético, tais como número de gerações, quantidade de indivíduos (população), taxa de *crossover* e de mutação, tamanho mínimo do cromossomo, índices de bônus e de penalidade da função de avaliação. Já os monitores e *plots* (gráficos) apresentam ao usuário informações pertinentes ao modelo. Neste caso, através dos monitores POPULACAO e GERACOES, é possível visualizar a quantidade de indivíduos do modelo e o progresso das gerações do algoritmo genético durante a execução da simulação. No *plot*, pode-se acompanhar o valor da função de *fitness* da melhor e da pior da solução através do tempo.

E por fim, têm-se os botões *switch* (chaves de liga-desliga), responsáveis por ativar ou desativar determinada característica do modelo. No modelo constam dois botões *switch*, ELITISMO? e USAR-FUZZY?. O primeiro controla a presença ou não de elitismo no algoritmo genético. Em caso afirmativo, o melhor indivíduo de cada geração é sempre repassado à geração seguinte. Se o elitismo estiver desativado, existe a possibilidade do melhor indivíduo de sua espécie em uma determinada época não ser repassado à próxima geração. Já o *switch* USAR-FUZZY? é responsável por ativar e desativar o uso da função *fuzzy* no recálculo da rota do robô.

2.4. As Funcionalidades

Na aba *procedures*, encontram-se todas as funcionalidades desenvolvidas. Para facilitar a compreensão do modelo, apresenta-se na Figura 2 um algoritmo representando as funcionalidades desenvolvidas no projeto.

O algoritmo inicia com a geração da população inicial do algoritmo genético. A quantidade de indivíduos da população é determinada pelo *slider* POPULACAO, que se encontra na interface do modelo. Cada indivíduo da população é gerado com a variável denominada *cromossomo*. Este *cromossomo* é um vetor binário que representa uma sequência de movimentos, onde cada movimento é codificado em três bits (necessários para codificar as oito possibilidades de movimentação) da seguinte forma: direita (000), acima-direita (001), acima (010), acima-esquerda (011), esquerda (100), abaixo-esquerda (101), abaixo (110) e abaixo-direita (111). Por exemplo, um *cromossomo,* que representa um percurso com 15 movimentos, composto por 15 genes, tem como fenótipo 001001001001000001001001010010011001001001000.

Os *cromossomos* podem possuir tamanhos distintos, iniciando como o tamanho determinado pelo *slider* MIN-TAM-CROMOSSOMO. Os movimentos que compõem os *cromossomos* também são escolhidos aleatoriamente. Logo em seguida, são realizados os movimentos presentes em cada cromossomo e calculado o valor da função de avaliação, também conhecida como *fitness*, de cada indivíduo.

A função de avaliação é responsável por quantificar a qualidade de cada indivíduo como uma possível solução para o problema proposto. Em um problema de maximização, quanto maior for o valor calculado pela função de avaliação, melhor será a solução encontrada. O problema do planejamento de rotas de robôs móveis consiste em um problema de minimização, pois se almeja determinar a melhor rota de deslocamento de um robô a partir de um ponto inicial até um estado objetivo. Portanto,

a melhor rota é determinada pela rota mais curta, ou seja, o menor cromossomo, desde que evite as colisões com os obstáculos.

```
Algoritmo

Gere a população de indivíduos aleatoriamente
Realize os movimentos presentes em cada cromossomo
Avalie o fitness de cada indivíduo ao final do trajeto

Repita o processo de AG pelo número de gerações
    Repita o processo de crossover de acordo com a taxa definida
        Selecione os pais através da "seleção por torneio"
        Divida em dois, numa posição randômica, cada um dos cromosssomos dos pais
        Gere dois novos indivíduos com a mescla do cromossomo dois pais

    Se o elitismo estiver ativado
        Selecione o melhor indivíduo
        Repasse-o para a geração subsequente

    Copie os demais indivíduos para a geração seguinte
    Elimine a população antiga
    Realize a mutação de acordo com a taxa definida
    Realize os movimentos presentes em cada cromossomo
    Avalie o fitness de cada indivíduo ao final do trajeto

Faça enquanto o menor fitness for >= 0
    Repita os mesmos processos do AG
    Adicione cromossomos randômicos a cada 10 gerações

Imprima a rota do melhor indivíduo
```

Figura 2. Pseudocódigo do Modelo

Para calcular o *fitness*, são executados todos os movimentos presentes no *cromossomo* de cada indivíduo. A cada movimento, verifica-se a presença de obstáculos pelo caminho, penalizando aqueles indivíduos cuja rota colide com os obstáculos. O procedimento funciona da seguinte forma: ao mover-se para um novo *patch*, de acordo com o movimento estipulado pelo *cromossomo*, é verificada a presença de obstáculos neste local. Em caso afirmativo, a candidata à solução de melhor rota é penalizada na função de avaliação de acordo com o valor estipulado na interface através do *slider* de penalidade.

Caso não haja obstáculos no *patch* selecionado para o movimento, uma nova verificação é realizada, mas desta vez relacionada ao estado objetivo. Se o *patch* escolhido for o estado objetivo a ser alcançado pela rota, a solução é bonificada (o valor desta bonificação é determinado pelo *slider* BONUS na interface) em sua função de avaliação. E finalmente, se o *patch* escolhido for apenas um *patch* comum, sem obstáculos ou estado objetivo presente, o valor da função de avaliação é apenas acrescido do valor de deslocamento do movimento.

Abaixo encontra-se a função de avaliação adotada para o problema:

$$\text{Fitness} = \sum_{i=0}^{n} distância * (1 + peso) + (15 - posiçãox + 15 - posiçãoy)$$

onde: *distância* é o trecho percorrido no movimento (movimentos para frente, para trás ou para os lados consomem uma unidade de distância; já os movimentos diagonais consomem $\sqrt{2}$ unidades de distância); *peso* é o valor atribuído pela penalização ou bonificação (este valor será zero caso nenhuma das opções anteriores seja atribuída à variável); *posiçãox* e *posiçãoy* indicam a posição atual da solução em relação aos eixos x e y.

Após a realização dos movimentos presentes nos cromossomos e o cálculo da função de avaliação de todos os indivíduos da população inicial, dá-se início ao processo do algoritmo genético e de seus operadores de *crossover* e mutação. Com base na taxa de *crossover*, determinada pelo *slider* CROSSOVER, são gerados novos indivíduos através da reprodução ou cópia (ex.: Se a taxa de *crossover* estiver marcando 70, isto significa que 70% da nova população será gerada através da reprodução e os outros 30% serão meramente uma cópia dos indivíduos da atual população).

O processo de reprodução empregado é o da seleção por torneio. Neste processo, são selecionados aleatoriamente três indivíduos da população atual. O indivíduo com o melhor *fitness* dentre os três é escolhido como um dos *pais*. O mesmo procedimento é executado para a escolha do outro *pai* do novo individuo. Em seguida, o código genético (*cromossomo*) de cada um dos pais selecionados é dividido aleatoriamente em duas partes, num processo intitulado de *crossover* de um ponto.

Na sequência, dois novos indivíduos são gerados e parte do *cromossomo* de cada progenitor é adicionada ao *cromossomo* do novo indivíduo. Neste caso, o primeiro indivíduo gerado a partir da reprodução armazenará em seu *cromossomo*, a primeira parcela dos genes de seu pai e a segunda parcela dos genes de sua mãe. Por sua vez, o segundo indivíduo gerado herdará a segunda parte dos genes de seu pai e a primeira parte dos genes de sua mãe.

Ao término do processo de reprodução, os demais indivíduos da nova população são gerados por clonagem, ou seja, selecionados através do mesmo processo de seleção por torneio empregado no *crossover*, porém sem o cruzamento entre dois indivíduos. Sendo assim, o indivíduo escolhido é apenas copiado para a geração seguinte. Se o elitismo estiver ativado através do *switch* ELITISMO? presente na interface, o melhor indivíduo da população atual é copiado para a geração subsequente. Com a nova população criada, a população antiga é eliminada e o processo de mutação tem início.

A mutação é efetuada com base na taxa de mutação determinada pelo *slider* TAXA-DE-MUTACAO (ex.: se o slider estiver posicionado em 10, as chances de ocorrer mutação no indivíduo são de 10%). O processo de mutação consiste na substituição de um dos genes do *cromossomo*, ou seja, a alteração de um dos movimentos da rota do indivíduo. Escolhe-se randomicamente um dos genes do cromossomo e o substitui por outro gene, também escolhido arbitrariamente, que será composto por um dos oito movimentos possíveis.

Com o fim do processo de mutação, todos os indivíduos da população são submetidos à movimentação (através das rotas presentes em seus cromossomos) e ao cálculo da função de avaliação. O processo do algoritmo genético repete-se até que o

número de gerações determinado na interface seja atingido. Caso o estado objetivo não seja alcançado através da melhor rota obtida ao final do processo, a simulação continua sendo executada, adicionando a cada 10 gerações, novos indivíduos gerados aleatoriamente, com o objetivo de agregar diversificação à população e fugir de máximos locais.

Uma vez que o estado objetivo é alcançado na melhor solução, executa-se a rota final do robô. Os movimentos executados pelo robô são alternados com os movimentos realizados pelos obstáculos, que por sua vez, são acionados através de outra função. Os obstáculos movimentam-se como uma espécie de serpente, onde sua cabeça é identificada pela cor marrom e determina a direção do movimento dos obstáculos, sendo seguidas pelas demais partes do corpo. A cabeça seleciona um dos *patches* vizinhos (vizinhança de von Neumann) para deslocar-se. Se o *patch* selecionado já estiver sendo ocupado por um agente, a função é chamada novamente para a escolha de um novo vizinho. A função é executada recursivamente até que um vizinho disponível seja encontrado. Satisfeita esta condição, a cabeça move-se para este *patch* e o restante dos obstáculos acompanham o movimento.

Com o deslocamento aleatório dos obstáculos móveis e a presença de obstáculos fixos no ambiente, a rota adotada pelo robô pode não alcançar o estado objetivo, visto que o cenário muda a cada passo. Sendo assim, é necessário realizar uma verificação constante da proximidade dos obstáculos com a rota em questão. Esta verificação pode ser realizada com ou sem o uso de uma função *fuzzy*. Ao utilizar-se da função *fuzzy*, dá-se ao robô autonomia de antecipar a decisão de recálculo de rota, de acordo com algum grau de pertinência para situação de colisão iminente. O robô pode perceber até quatro patches à sua frente. Caso a função detecte a presença de obstáculos dentro deste perímetro de até quatro patches, um grau de pertinência para colisão é calculado e utilizado como a probabilidade de recálculo da rota do robô. Se os obstáculos forem detectados a uma distância de quatro patches do robô, a probabilidade de recálculo é de 40%; a uma distância de três patches, a probabilidade é de 60%; a uma distância de dois patches, a probabilidade é de 80%; a uma distância de um patch, a probabilidade é de 100% (neste caso, o recálculo é obrigatório, visto que o próximo movimento do robô irá fazê-lo colidir com o obstáculo).

Ao não utilizar a função *fuzzy* para detectar a presença de obstáculos no entorno da rota, o recálculo da mesma só é efetuado quando o obstáculo estiver no *patch* subsequente em que se encontra o robô. Independentemente da abordagem empregada, com ou sem *fuzzy*, o recálculo da rota irá sempre ocorrer para evitar a colisão do robô com os obstáculos.

No recálculo da rota, são realizados os mesmos procedimentos descritos anteriormente, mas desta vez, em um cenário completamente diferente. O recálculo da rota pode ser executado diversas vezes ao longo do trajeto do robô – a cada novo obstáculo encontrado pelo caminho, um novo recálculo de rota deverá ser efetuado. Desta forma, evitam-se as colisões e obtêm-se a melhor rota para o robô. Ao atingir o estado objetivo, o *cromossomo* é editado, removendo-se todos os genes excedentes que se encontram após o gene que atingiu o estado objetivo (ex.: se um *cromossomo*

composto por 25 genes atingiu o estado objetivo no gene 23, os genes 24 e 25 são descartados).

Durante a simulação, são impressos no gráfico os valores da função de avaliação do melhor e o pior indivíduo da população de cada em função do tempo. Enquanto o gráfico gerado pelos piores indivíduos ao longo das gerações apresenta um aspecto altamente oscilatório, com grandes variações no valor da função de avaliação, o gráfico gerado pelos melhores indivíduos ao longo do tempo exibe um decrescimento no valor da função de avaliação, por vezes atingindo uma estabilidade em determinado ponto da simulação.

3. Simulações e Resultados

Para avaliar a viabilidade da proposta de planejamento de rotas de robôs móveis utilizando a abordagem de algoritmos genéticos, utilizou-se o algoritmo A* (A *star*) para a realização de testes de desempenho comparativos. O algoritmo A* foi desenvolvido por Peter Hart, Nils Nilsson e Bertram Raphael em 1968 e é amplamente utilizando em problemas de planejamento de rotas e grafos transversais, devido a sua alta *performance* – é um algoritmo completo e ótimo (RUSSEL e NORVIG, 2003).

O algoritmo genético utilizado nos testes de desempenho foi configurado da seguinte forma: população de 299 indivíduos, 80% de taxa de *crossover*, 20% de taxa de mutação e elitismo ativado. Durante a simulação, o AG é executado por pelo menos 38 gerações – caso o estado objetivo não seja alcançado nas 38 primeiras gerações, o modelo segue sendo executado até que o robô consiga atingir a sua meta.

Tabela 1. Medição da quantidade de movimentos realizados nas rotas

Cenários					
Sem Obstáculos		**Obstáculos Fixos**		**Obstáculos Fixos e Móveis**	
A*	**AG**	**A***	**AG**	**A***	**AG**
15	16	15	21	18	17*
15	18	15	30	21*	25****
15	16	15	21	18	25*
15	16	15	21	16*	22*
15	18	15	19	20**	23
15	16	15	20	16*	25
15	18	15	17	18	20***
15	20	15	21	18	23
15	15	15	18	17*	21
15	16	15	26	18	24
15	15	15	20	18	21
15	16	15	24	21*	20
15	16	15	20	21*	24*

15	19	15	24	18	29
15	16	15	20	18	25
15	15	15	21	18	30
15	17	15	18	18	25******
15	16	15	20	18*	20*****
15	15	15	16	18	19*
15	16	15	19	18	25

Ao todo, foram realizados vinte simulações em três cenários distintos: um cenário sem obstáculos, um cenário com obstáculos fixos e um cenário com obstáculos fixos e móveis. Como parâmetro de comparação de desempenho dos dois modelos, fez-se uso do tamanho da rota – número de movimentos empregados pelo robô para atingir o estado objetivo. Na Tabela 1, apresentam-se os resultados obtidos durante os testes de desempenho comparativos entre o A* e o AG.

Para avaliação dos testes de desempenho realizados utilizou-se o Teste t de Student. Nos testes realizados com os três cenários elaborados, o A* apresentou uma diferença estatística extremamente significante em relação ao AG, ou seja, há 99,9% de certeza de que o A* foi superior ao AG nos testes realizados. Comparando apenas os resultados obtidos nos testes de desempenho onde houve o recálculo da rota em ambientes com obstáculos fixos e móveis (os asteriscos na tabela de resultados representam o número de recálculos realizados em cada rota), a diferença entre os dois modelos é estatisticamente significante, ou seja, 95% de certeza.

Apesar dos resultados adversos apresentados pela abordagem do AG em relação ao A*, há um indicativo de melhora nos resultados do AG à medida que cresce a complexidade do cenário. Por exemplo, no cenário 3 (com obstáculos fixos e móveis) as rotas calculadas pelo AG foram, em média, 26% maiores do que as rotas calculadas com o A*. Entretanto, ao serem levadas em consideração apenas as situações em que houve recálculo de rota, a rotas calculadas pelo AG foram, em média, 19% maiores. Isso leva a crer que em cenários maiores (nos testes o cenário era de tamanho 15x15), com maior quantidade de obstáculos e, por conseguinte, com maior probabilidade de colisão, o modelo AG emparelharia com o A* na qualidade da rota calculada.

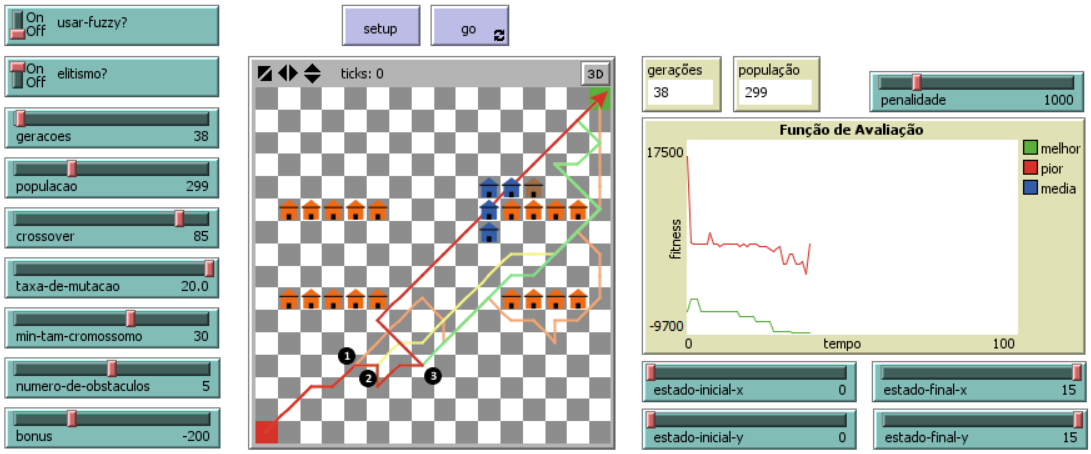

Figura 3. Solução final gerada através do AG após três recálculos de rota

No exemplo demonstrado pela Figura 3, pode-se constatar o encontro do robô com os obstáculos durante a execução de sua rota em três pontos distintos – pontos pretos 1, 2 e 3. Nestes momentos, a rota é recalculada para evitar a colisão do robô com os obstáculos. A cada recálculo de rota, o robô encontra um novo caminho livre de obstáculos. No entanto, nos dois primeiros recálculos, os obstáculos acabaram cruzando o caminho do robô novamente, forçando-o a buscar uma rota alternativa. Após o terceiro recálculo, o robô encontrou uma trajetória e alcançou o estado objetivo sem colidir com nenhum obstáculo.

4. Conclusões

Após o desenvolvimento do modelo proposto e da análise dos resultados obtidos, conclui-se que a abordagem de algoritmos genéticos para o problema de planejamento de rotas de robôs móveis em ambientes estáticos e dinâmicos apresentou-se com uma alternativa viável para a solução do problema em questão.

Através da abordagem de algoritmos genéticos, foi possível modelar soluções para o problema através dos *cromossomos* compostos pelos movimentos do robô no ambiente. A partir de soluções geradas arbitrariamente, os processos de *crossover* e mutação, baseados na função de avaliação de cada indivíduo, foram capazes de evoluir as candidatas à solução do problema, atingindo uma solução satisfatória para cada uma das simulações.

A utilização do *software* Netlogo também foi de suma importância no desenvolvimento do modelo proposto, pois possibilitou a modelagem de um ambiente dinâmico, composto por obstáculos móveis, onde o robô é capaz de detectar os obstáculos e evitar possíveis colisões. Através da interação entre os agentes do ambiente, o robô pode perceber os obstáculos em seu caminho e recalcular a sua trajetória para alcançar o estado objetivo.

O modelo de AG empregado neste trabalho ainda é um modelo em estágio seminal, que faz uso apenas dos operadores básicos de *crossover* e mutação. Conforme foi explanado na introdução do trabalho, o algoritmo genético é uma poderosa ferramenta capaz de ser combinada com diferentes técnicas para aprimorar o seu desempenho, tais como Redes Neurais de Hopfield e Campo de Potencial Numérico. Além disso, é possível acrescentar novos operadores mais robustos, capazes de elevar a *performance* do modelo.

Apesar da constatação da superioridade do A* em relação ao AG nos testes desempenho realizados, pode-se esperar uma melhora no modelo de AG em modelos mais complexos – cenário com obstáculos fixos e móveis envolvendo o recálculo da rota. Os cenários empregados nos testes favorecem o modelo do A*, pois este é um algoritmo ótimo. No entanto, este tipo de algoritmo tende a ter seu desempenho reduzido em ambientes mais complexos. Desta forma, em trabalhos futuros, pretende-se aperfeiçoar o modelo de AG com a adição de novos operadores e técnicas de refinamento, submetendo os modelos utilizados a testes de desempenho em ambientes maiores e mais complexos, com um número superior de obstáculos fixos e móveis, além da adição de outros robôs no ambiente, caracterizando um ambiente multiagente.

Agradecimentos

Os autores deste trabalho agradecem à CAPES pelo auxílio na forma de bolsa de mestrado REUNI. Este trabalho é parcialmente financiado pelo CNPq (Proc. 476234/2011-5, 560118/10-4,305131/2010-9) e FAPERGS (Proc. 11/0872-3).

Referências

AGUIAR, M. Análise Formal da Complexidade de Algoritmos Genéticos. Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, Brasil, 1998. (Dissertação de Mestrado)

CONCILIO, R. Contribuições à Solução de Problemas de Escalonamento pela Aplicação Conjunta de Computação Evolutiva e Otimização com Restrições. Universidade Estadual de Campinas, Campinas, SP, Brasil, 2000. (Dissertação de Mestrado)

HU, Y.; YANG, S. X. A knowledge Based Genetic Algorithm for Path Planning of a Mobile Robot. Nova Orleans, EUA, 2004.

LI, Q.; TONG, X.; XIE, S.; ZHANG, Y. Optimum Path Planning for Mobile Robots Based on a Hybrid Genetic Algorithm. China, 2006.

LIN, L.; WANG, H.; WU, Q. Improved Genetic Algorithms Based Path Planning of Mobile Robot Under Dynamic Unknown Environment. Louyang, China, 2006.

RUSSEL, S. J.; NORVIG. Artificial Intelligence: A Modern Approach. Reading: Prentice Hall, 2003.

SADATI, N.; TAHERI, J. Genetic Algorithm in Robot Path Planning in Crisp and Fuzzified Environments. Tehran, Irã, 2002.

TU, J.; YANG, S. X. Genetic Algorithm Based Path Planning for a Mobile Robot. Taipei, Taiwan, 2003.

WILENSKY, URI. Guia do Usuário: Netlogo. Disponível em http://ccl.northwestern.edu/netlogo/docs. Acessado em 21 de Outubro de 2011.