Proceedings of the IV Brazilian Conference on Neural Networks - IV Congresso Brasileiro de Redes Neurais
pp. 118-123, July 20-22, 1999 - ITA, São José dos Campos - SP - Brazil

SNNAP – Sistema Neural de Navegação em Ambientes Pré-Mapeados

Jerusa Marchi Vaz, João Alberto Fabro
Departamento de Ciência da Computação-Centro de Ciências Exatas e Tecnológicas
UNIOESTE - Universidade Estadual do Oeste do Paraná
Campus de Foz do Iguaçu
Av. Tancredo Neves, Estrada de Acesso a Furnas, Km 1,3
Caixa Postal 961 CEP 85855-000
E-mails: jerusa@dcc.unioeste-foz.br, jfabro@dcc.unioeste-foz.br

Abstract

In this paper is presented a proposal of an hibrid control system for navigation of mobile vehicles, that uses sensor based navigation together with map-based trajectory planning. The resulting system is named SNNAP (Sistema Neural para Navegação em Ambientes Pré-Mapeados - Neural System for Navigation in Known Environments). This system integrates Neural Networks for avoiding unknown obstacles, and a map-based algorithm for trajectory planning. Are also presented studies of previous proposals, a mobile vehicle simulator named Khepera, in which this approach has been tested, and the steps taken to develop the system.

1. Introdução

A navegação autônoma de veículos há muito desperta o interesse de pesquisadores, principalmente na área de inteligência artificial, sendo que o problema básico desta pesquisa concentra-se na determinação de uma trajetória, de modo que o veículo possa navegar pelo ambiente evitando colidir com obstáculos.

A complexidade envolvida no problema da navegação autônoma está relacionada com as muitas posições distintas do veículo no ambiente, sendo que, devido a isto, o sistema de controle deve ser capaz de reconhecer e tratar cada situação específica, ou possuir capacidades de generalização .

Várias são as propostas para a solução deste problema, sendo o planejamento de trajetória (abordagens baseadas em modelos) e a navegação utilizando informações fornecidas por sensores (abordagens baseadas em sensores) as duas classes de abordagens principais [1].

Neste trabalho é estudada experimentalmente uma proposta de integração entre a abordagem de planejamento de trajetória baseada em um pré-mapeamento do ambiente, e a abordagem baseada em sensores que busca o desvio de objetos dinâmicos encontrados durante a execução da trajetória.

São apresentados os estudos realizados para o desenvolvimento do projeto, um simulador de robôs autônomos, e a proposta para o controle da navegação de robôs em ambientes estáticos com objetos dinâmicos. São apresentadas as análises dos resultados obtidos em simulações, além de algumas possíveis aplicações.

2. Estudos realizados

Para o desenvolvimento deste projeto realizaram-se estudos sobre:

- a teoria das redes neurais artificiais
- funcionamento do simulador de um robô móvel, denominado Khepera [2], e
- propostas de controle para navegação autônoma com redes neurais artificiais

2.1. Redes Neurais Artificiais

A idéia de replicar um cérebro por meios puramente técnicos acompanha as civilizações desde a antigüidade. Muito antes de se ter um conhecimento mais detalhado do sistema nervoso, teorias mecanicistas surgiram buscando simular comportamentos inteligentes [3].

As redes neurais artificiais caracterizam o modelo mecanicista *bottom-up*, que surgiu a partir das observações experimentais do potencial de ação e de algumas topologias em que neurônios se interligavam formando redes, onde Warren McCulloch apresenta-se como sendo um dos primeiros a gerar um destes modelos [4].

Também denominada teoria conexionista, as redes neurais artificiais possuem como vantagens o controle altamente distribuído e paralelo[5], com elevada imunidade à ruídos, além da capacidade de "aprender" através de exemplos e generalizar o aprendizado, demonstrando bom desempenho em tarefas mal definidas .

As redes neurais artificiais são constituídas por elementos básicos de processamento, conectados entre si, denominados neurônios artificiais. Cada neurônio possui uma única saída, que pode estar conectada a muitos outros neurônios, formando redes. As redes geralmente possuem uma camada de entrada, uma

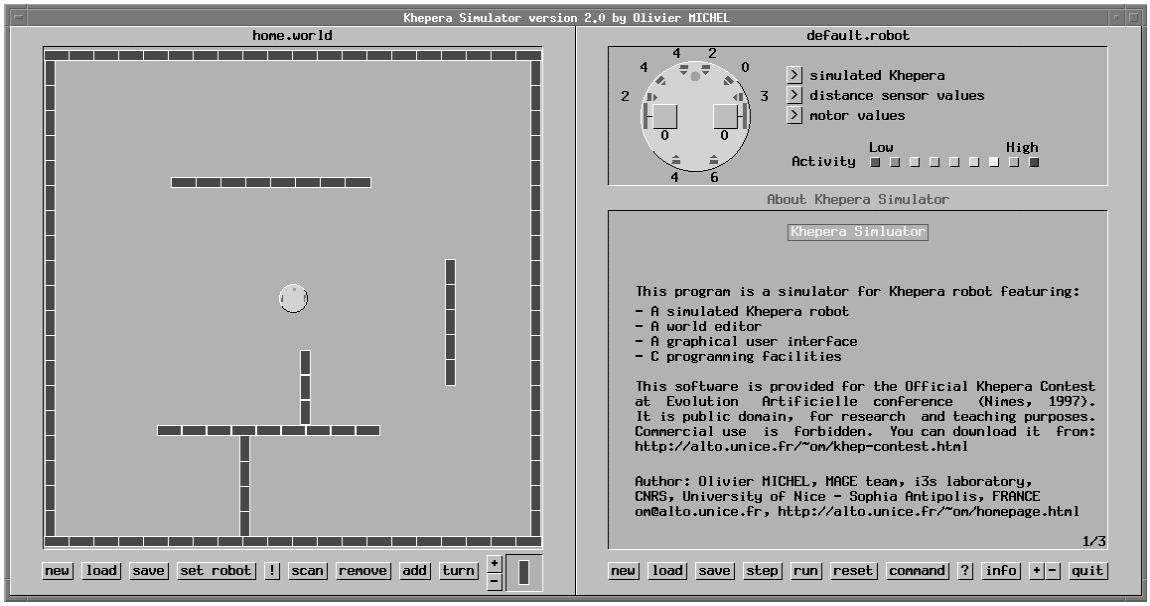

Figura 1 - O simulador Khepera

camada de saída e camadas intermediárias, chamadas de camadas ocultas.

O conhecimento de uma rede neural está armazenado em suas sinapses, que são as conexões com outros neurônios; desta forma, ao se modificar os pesos destas conexões, atualiza-se o conhecimento da rede. A este processo dá-se o nome de treinamento [6].

As redes neurais podem ser classificadas quanto a topologia das conexões entre os neurônios e quanto ao tipo de treinamento utilizado. Quanto a topologia das conexões, as redes podem ser classificadas em recorrentes ou não recorrentes.

Nas redes neurais recorrentes os neurônios são realimentados, ou seja, a saída do ciclo atual é reaplicada no ciclo seguinte como sendo a nova entrada da rede. Este processo é repetido até a estabilização da rede. Já as redes neurais não recorrentes não possuem conexões de realimentação, suas conexões sempre partem da camada de entrada em direção a camada de saída.

Quanto ao tipo de treinamento, elas podem ser classificadas em redes com treinamento supervisionado e redes com treinamento não supervisionado.

O treinamento supervisionado é o mais utilizado e exige a disponibilidade de um conjunto de treinamento formado por pares de vetores de entrada e saída e o algoritmo mais utilizado para isto é denominado Retropropagação (*BackPropagation*). Este algoritmo consiste basicamente em retropropagar o erro gerado na saída da rede, de forma a estabilizar os pesos das conexões .

No treinamento não supervisionado, o conjunto de treinamento consiste somente de vetores de entrada. O algoritmo utilizado é denominado aprendizado competitivo ou aprendizado de Kohonen. Por este princípio, os neurônios vão sendo agrupados segundo sua similaridade com o vetor de entrada, formando categorias. Desta forma a rede estará treinada quando todos os vetores representantes das categorias forem classificados corretamente pela rede [3].

2.2. O simulador Khepera

O simulador Khepera é um programa de domínio público escrito por Olivier Michel[2], que permite a implementação de algoritmos de controle usando as linguagens C ou C++. Seu funcionamento se dá em estações Unix, usando, para gerar a interface, a biblioteca gráfica X11.

Este simulador se divide em duas partes: o "mundo", que simula um ambiente de $1m^2$, ocupando a parte esquerda da tela, e o "robô", que simula o robô Khepera real (que tem 5 cm de diâmetro), na parte direita da tela (Figura 1). Na parte do mundo pode-se observar o desempenho do robô enquanto este navega pelo ambiente, e na parte do robô pode-se visualizar os dados do robô (valor dos sensores e motores), além de outras informações pertinentes.

2.2.1. Descrição do mundo. O simulador Khepera disponibiliza diversos modelos de mundo, os quais podem ser carregados através do botão *load* que se encontra na parte esquerda da tela. Além dos mundos disponíveis, o Khepera permite que novos mundos sejam criados, através dos botões *new* e *save*. Um mundo é composto por objetos, como tijolos e lâmpadas, que podem ser selecionados pelos botões + e - e adicionados ou removidos do mundo pelos botões *add* e *remove*. Os tijolos também podem ser rotacionados sobre seu próprio eixo pressionando o botão *turn*. Para que o robô reconheça os objetos no ambiente é necessário pressionar o botão *scan* e para saber o que o robô está percebendo basta pressionar o botão *!*.

2.2.2. Descrição do robô. O robô Khepera é composto por 8 sensores infravermelhos que, por reflexão, detectam a proximidade dos objetos; 8 sensores de luz que medem o nível da luz no ambiente; e dois motores que possibilitam a navegação do robô no ambiente.

A posição do robô no mundo pode ser estipulada através do botão *set robot*, que permite colocar o robô em qualquer parte do ambiente; também é possível orientar a direção do robô pressionando-se o botão *command* e digitando na linha de comando *set angle* e o ângulo desejado, por exemplo : *set angle 45*.

2.3. Propostas de Controle para a Navegação Autônoma com Redes Neurais Artificiais

O uso de redes neurais artificiais em abordagens baseadas na leitura de sensores possibilita o treinamento da navegação do veículo, diretamente a partir da leitura dos sensores (cada leitura de sensor representa uma entrada para um neurônio da rede). Deste modo, é possível definir uma estratégia de controle a partir de exemplos de navegação, não sendo necessária a definição de regras rígidas, levando a generalização das reações do veículo perante as várias situações encontradas no ambiente. Além disto, possibilita uma maior eficiência no processamento das leituras dos sensores, permitindo interação com o ambiente em tempo real. Uma proposta estudada, denominada "Navegador em Redes Neurais", foi desenvolvida pelos alunos Dieferson Luis Alves de Araujo e Edgar Noda da Universidade Federal do Paraná [7].

O trabalho consiste em um estudo da aplicação de redes neurais artificiais à navegação autônoma de robôs. Após várias experiências, o trabalho resultou em duas redes neurais com topologia não recorrente e treinamento supervisionado para controle do robô Khepera. Cada rede possui 9 entradas (8 sensores do robô + um *bias term*), 27 neurônios na camada oculta e 2 na camada de saída (controle dos motores). Uma das redes é responsável pelo desvio dos obstáculos e a outra por buscar alvos pré-estabelecidos no ambiente (luzes).

As redes neurais de luzes e colisões são integradas por um programa simples que prioriza a rede de colisões, desta forma quando há possibilidade de colisão a rede de luzes não é utilizada, e somente ao se afastar do objeto com o qual havia possibilidade de colisão, a rede de luzes é reativada. Este controle faz com que o robô navegue aleatoriamente pelo ambiente sem colidir com obstáculos, até que seus sensores de luminosidade sejam acionados, fazendo com que o robô vá em direção a luz.

Quando não há nenhuma luz no raio de ação dos sensores, e nenhum obstáculo próximo, o robô anda para frente. Deste modo, não há nenhum tipo de controle global sobre a sua trajetória, e muitas vezes ele não alcança as luzes, pois identifica a luz atrás de um obstáculo, desvia do obstáculo e os sensores de luz não mais percebem a luminosidade. A figura 2 apresenta um exemplo de execução do Navegador em Redes Neurais.

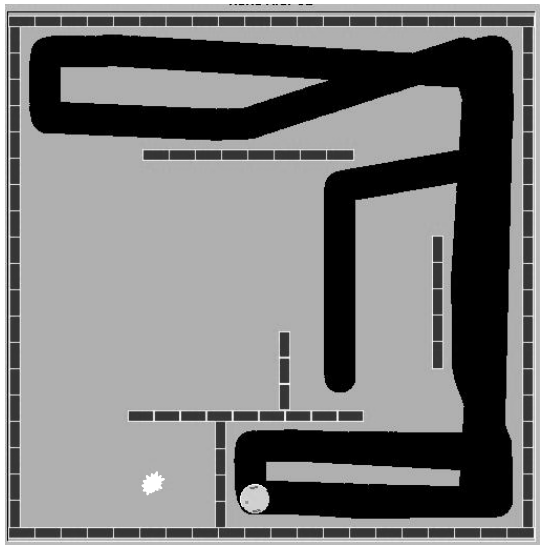

Figura 2-Exemplo do Navegador em Redes Neurais

3. Desenvolvimento do projeto

O SNNAP é dividido em dois níveis, o primeiro destina-se ao mapeamento do ambiente e ao controle da trajetória do veículo. O segundo tem por objetivo desviar o veículo de possíveis obstáculos que não foram previamente mapeados.

O desenvolvimento do SNNAP baseou-se no Navegador em Redes Neurais. O primeiro passo foi fazer uma estimativa da luz no ambiente, como forma de solucionar o problema da perda da luminosidade pelos sensores.

Em seguida, o mapeamento do ambiente e o planejamento da trajetória do veículo foram incluídos, caracterizando a abordagem baseada em modelos e o primeiro nível do sistema. A inclusão de obstáculos não mapeados no caminho do veículo e o uso da rede neural de colisões para o desvios destes obstáculos caracteriza a abordagem baseada em sensores e o segundo nível do sistema, integrando assim, as duas abordagens.

3.1. Estimativa da Luz no Ambiente

Visando melhorar o desempenho do robô na busca por luzes no ambiente, optou-se por criar uma estimativa da luz. Desta forma, ao navegar pelo ambiente, quando o robô identifica luminosidade, é criada uma estimativa desta luz. Ao encontrar um obstáculo, o robô o desvia e verifica se há alguma luz estimada, buscando-a. A figura 3 mostra o resultado da aplicação da estimativa da luz em um caso onde somente o Navegador em Redes Neurais não conseguia alcançá-la.

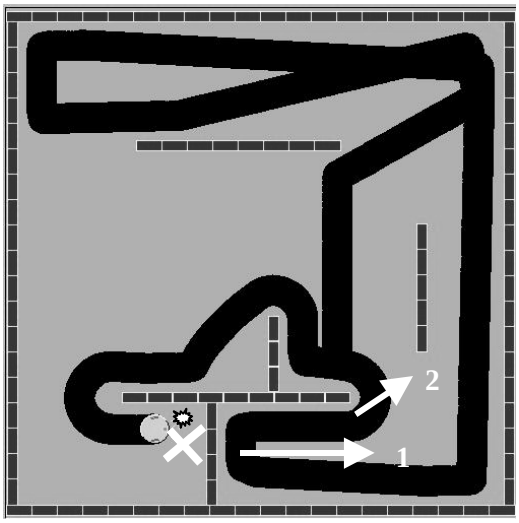

Figura 3 - Estimativa da luz no Ambiente

Em (1) o robô identifica a luz e faz a estimativa(X), que é baseada no ângulo do robô no ambiente e no sensor de luz mais ativo. Após desviar-se do obstáculo (2), a estimativa da luz é acionada, ativando os sensores de luminosidade do robô, através de entradas "falsas" na rede de luzes, fazendo-o se dirigir para a estimativa da luz (X). Quando o robô se aproxima do ponto estimado, os sensores identificam novamente a luminosidade e a rede de luzes volta a receber como entrada a leitura real dos sensores, abandonando a estimativa.

Este procedimento permitiu um desempenho mais eficiente do robô na busca por luzes no ambiente, pois mesmo que a rede de colisões seja acionada, o robô mantém a estimativa da luz, buscando por este ponto ao se distanciar dos obstáculos que poderiam provocar colisão.

3.2. Implementação do 1º Nível: Mapeamento do Ambiente e Planejamento da Trajetória

O objetivo da implementação deste primeiro nível é permitir que o robô chegue a um ponto determinado (denominado ponto objetivo), alcançando pontos pré-estabelecidos intermediários (luzes falsas), evitando colisões. A figura 4 apresenta um exemplo do comportamento do robô com o mapeamento do ambiente e a utilização dos falsos pontos de luz.

3.2.1.Mapeamento do Ambiente: O objetivo desta etapa é armazenar dados sobre o ambiente. A partir destas informações, será possível planejar a trajetória do robô. Após analisar várias formas de mapeamento, optou-se por percorrer a lista de objetos do simulador, buscando pelos tijolos do ambiente, armazenando a posição destes em uma matriz 1000x1000, sendo 1 para existência de obstáculo e 0 para não existência de obstáculo em uma determinada posição. Desta forma cada tijolo de tamanho 25x50 é mapeado como uma seqüência de 1's a partir do ponto x e y do tijolo.

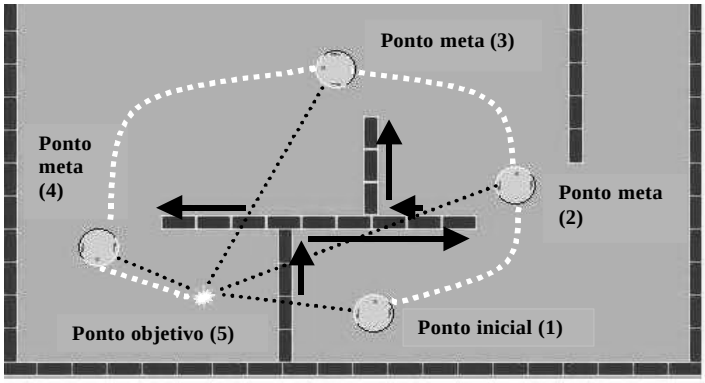

Figura 4 - Execução em um ambiente pré-mapeado

3.2.2. Planejamento da Trajetória: O planejamento da trajetória tem como objetivo encontrar um caminho que conduza o robô de seu ponto inicial até um ponto estabelecido como ponto objetivo, que é representado por uma luz no ambiente. Existem diversos tipos de abordagens para o planejamento da trajetória, dentre elas um algoritmo denominado "*on-line*"[8], que consiste em traçar uma reta do ponto inicial até o ponto objetivo. Esta reta é seguida pelo robô até que este encontre um obstáculo, então o robô tenta contornar o obstáculo e voltar aos pontos da reta. A idéia deste algoritmo foi utilizada na execução do planejamento, porém foram agregados à ela alguns aspectos que a tornassem mais eficiente. A figura 4 apresenta um exemplo de planejamento de trajetória para o mundo *home.world*. O algoritmo de planejamento traça, igualmente ao algoritmo *on-line*, uma reta do ponto inicial (1), que é a posição atual do robô, até o ponto objetivo (5), representado pela luz; contudo, se houver uma parede entre estes dois pontos, esta por sua vez é percorrida buscando localizar o ponto da borda que é estabelecido como ponto meta (2) do robô para alcançar o ponto objetivo. Este ponto meta é utilizado para a próxima iteração, novamente trançando uma reta do ponto meta (2) até o ponto objetivo (5), havendo parede entre estes dois pontos, busca-se por seu final (borda), marcando o ponto encontrado como ponto meta (3); este ponto meta é então usado como ponto inicial para a próxima iteração, e assim sucessivamente até que não hajam mais obstáculos entre o ponto inicial e o ponto objetivo. Ao buscar cada ponto meta, o robô executa a trajetória planejada, alcançando o ponto objetivo. A figura 5 apresenta outro exemplo de execução em um ambiente pré-mapeado com planejamento de trajetória.

3.3. Implementação do 2º Nível - Inclusão de Obstáculos Dinâmicos

São caracterizados como obstáculos dinâmicos tanto objetos estáticos que não foram previamente mapeados (mesas, cadeiras, e outros) como objetos dinâmicos que estejam presentes no ambiente no momento da execução (como pessoas e outros robôs móveis.

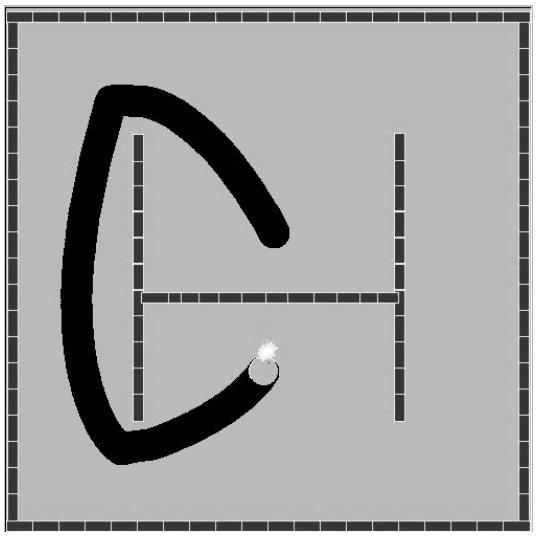

Figura 5 – Outro exemplo de execução com mapeamento do ambiente e planejamento de trajetória

A inclusão de obstáculos não mapeados foi feita através da inclusão de tijolos no ambiente, após seu mapeamento, através da função *scan*, que possibilita ao robô perceber os novos obstáculos, sem que estes estejam em seu mapa do mundo. As figuras 6 e 7 apresenta dois exemplos onde obstáculos não mapeados foram incluídos.

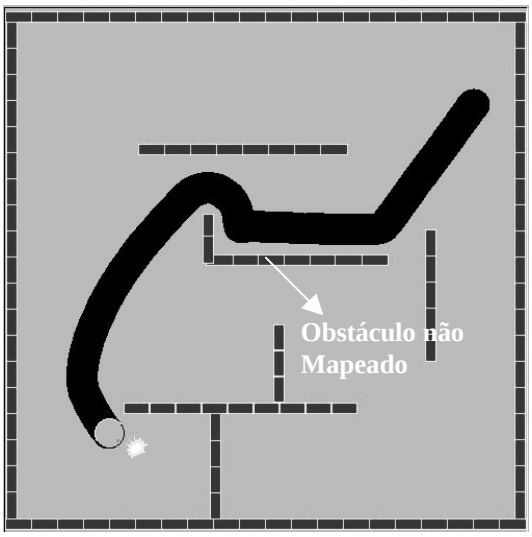

Figura 6 – Exemplo da navegação em ambientes pré-mapeados com obstáculos não mapeados.

Quanto a inclusão de objetos dinâmicos (outros robôs), utilizou-se uma biblioteca disponibilizada pelo simulador, denominada "*multirobots*". Esta biblioteca permite a criação de uma lista de robôs. Desta forma, fez-se um tratamento distinto do primeiro robô desta lista, o qual atende ao sistema SNNAP, enquanto o outro atende a um programa simples de desvio de obstáculos baseado na leitura dos sensores. A figura 8 apresenta as principais etapas executadas pelo robô para buscar pelo seu ponto objetivo, desviando de obstáculos dinâmicos.

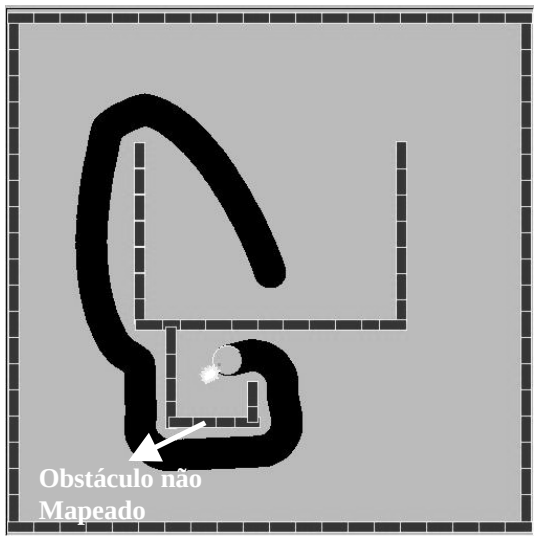

Figura 7 – Outro exemplo com obstáculos não mapeados.

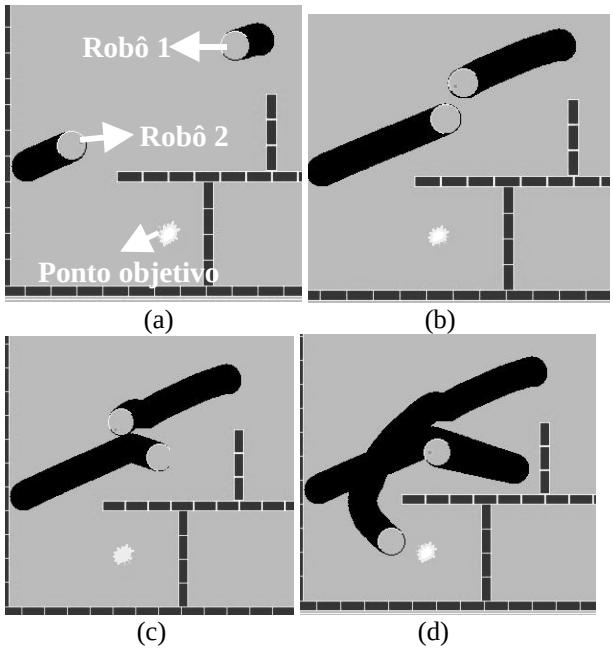

Figura 8 –Ambiente com obstáculo dinâmico.

Iniciada a execução (Figura 8 (a)), o robô controlado pelo sistema SNNAP (Robô 1) realiza o mapeamento do ambiente e o planejamento da trajetória, partindo em busca do ponto objetivo. Quando os sensores de contato indicam colisão (Figura 8 (b)), ocorre o desvio (Figura 8 (c)) e a retomada da busca pelo ponto objetivo (Figura 8 (d)).

3.4. Análise dos Resultados

3.4.1. Navegador em Redes Neurais : O Navegador em Redes Neurais apresenta desempenho satisfatório em ambientes que não possuam um elevado número de obstáculos, ou em ambientes do tipo labirinto, onde as paredes guiem o robô até o ponto objetivo (luz). No entanto, as localizações do ponto objetivo e do robô influenciam diretamente a eficiência da execução, fazendo com que muitas vezes o robô não consiga alcançar a luz. Isto se deve ao fato do uso único da leitura dos sensores. Esta abordagem é interessante para ambientes desconhecidos e com obstáculos dinâmicos, justamente pela sua característica de desvio destes obstáculos, porém, este desvio pode afastar o robô de seu ponto objetivo, tornando a execução ineficiente para uma grande gama de situações.

3.4.2. Estimativa da Luz : A inclusão de uma estimativa da luz na abordagem baseada em sensores permite que o robô alcance a luz em uma série de casos testados, porém o robô, em algumas situações percorre vários caminhos antes que os seus sensores tenham o primeiro contato com a luminosidade, o que faz com que a eficácia da execução seja afetada.

3.4.3. Mapeamento do Ambiente e Planejamento da Trajetória : O mapeamento do ambiente e o planejamento da trajetória permitem ao robô percorrer uma trajetória de forma eficaz para alcançar o ponto objetivo, reduzindo a quantidade de caminhos que o robô percorre para chegar até a luz. O planejamento de trajetória é interessante, quando o ambiente é conhecido e estático, pois faz com que o robô saiba o caminho que o leva até seu ponto objetivo. Porém, a aplicação desta abordagem isolada não se mostra eficiente quando há obstáculos que não foram previamente mapeados.

3.4.4. Inclusão de Obstáculos Dinâmicos : A integração das abordagens baseadas em sensores e em modelos possibilita ao robô navegar em ambientes conhecidos com obstáculos dinâmicos, melhorando seu desempenho. O robô conhece seu ponto objetivo e mesmo que precise desviar de outros obstáculos desconhecidos, consegue retomar sua trajetória e ir até a luz.

4. Aplicações do projeto

Um robô real controlado pelo SNNAP poderá ser utilizado para tarefas simples tais como o transporte de documentos entre repartições e o acesso a locais difíceis, como é o caso dos dutos de ventilação de edifícios, onde o robô poderia realizar a manutenção destes dutos. Outra aplicação para um robô controlado por este sistema, seria o acesso a locais que ofereçam risco para pessoas, como ambientes radioativos, por exemplo.

5. Conclusões e Perspectivas Futuras

O SNNAP – Sistema Neural de Navegação em Ambientes Pré-Mapeados, busca solucionar o problema da navegação autônoma, integrando as abordagens baseadas em modelos e em sensores, através do uso de redes neurais artificiais, de forma que um veículo possa navegar em um ambiente pré-mapeado, seguindo uma trajetória, porém desviando de possíveis obstáculos dinâmicos ou não mapeados.

O uso de mapas do ambiente possibilitam que o veículo encontre rapidamente o ponto objetivo, pois não navega aleatoriamente pelo ambiente, e o uso da leitura dos sensores permite a identificação de obstáculos durante a execução da trajetória, e com as redes neurais a generalização das reações do robô ao ambiente é conseguida.

Como propostas de trabalhos futuros, citam-se a criação de um mapa em tempo de execução, de forma que, mesmo em ambientes inicialmente desconhecidos, o desempenho máximo do robô possa ser obtido, e o desenvolvimento de um veículo em hardware, para o qual podem ser implementadas melhorias, que permitirão a expansão tanto da parte de hardware como a parte de software, como a inclusão de manipuladores e câmeras de vídeo. Atualmente este projeto está aguardando a aprovação da compra de um robô Khepera real, no qual possam ser validadas as pesquisas realizadas com o simulador, levando em conta erros de leitura de sensores, imprecisão do mapeamento e outras dificuldades.

6. Referências Bibliográficas

[1] Fabro, João Alberto; Gomide, Fernando; Grupos Neurais e Sistemas Nebulosos: Aplicação à Navegação Autônoma, FEE/UNICAMP, 1996. Tese de Mestrado .

[2] Michel, Oliver, Khepera Simulator version 2.0 – User Manual, 1996. http://wwwi3s.unice.fr/~om/khep-sim.html, .

[3] Kovács, Zsolt Lászlo, O Cérebro e Sua Mente: uma introdução à neurociência computacional, Edição Acadêmica, São Paulo, 1997.

[4] McCulloch, W.S., and Pitts W.; A logical calculus of the ideas imminent in nervous activity, Bulletin of Mathematical Biophysics, vol 5, 1943.

[5] Rich, Elaine, Knight, K.; Inteligência Artificial, Makron Books, São Paulo, 1993.

[6] Blum, Adam, Neural Networks in C++; An object-oriented framework for building connectionist systems, John Wiley & Sons, New York, 1992.

[7] Araujo, Dieferson L. A., Noda, E.; Navegador em Redes Neurais. UFPR, 1996.

[8] Russel, S. J., Norvig, P.; Artificial Intelligence: A Modern Approach. Prentice-Hall, New Jersey, 1995.