UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA MECÂNICA
MESTRADO EM ENGENHARIA MECÂNICA
SISTEMAS DINÂMICOS

ESDRAS SALGADO DA SILVA

DESENVOLVIMENTO DE ARQUITETURAS HÍBRIDAS ATRAVÉS DE SISTEMAS COMPUTACIONAIS INTELIGENTES APLICADOS À ROBÓTICA MÓVEL AUTÔNOMA

DISSERTAÇÃO DE MESTRADO

CORNÉLIO PROCÓPIO
2015

ESDRAS SALGADO DA SILVA

DESENVOLVIMENTO DE ARQUITETURAS HÍBRIDAS ATRAVÉS DE SISTEMAS COMPUTACIONAIS INTELIGENTES APLICADOS À ROBÓTICA MÓVEL AUTÔNOMA

Trabalho de dissertação apresentado como requisito para a obtenção do título de Mestre em Engenharia Mecânica, do departamento de Sistemas Dinâmicos do Programa de Pós-Graduação em Engenharia Mecânica, da Universidade Tecnológica Federal do Paraná.

Orientador: Prof. Dr. Márcio Mendonça

CORNÉLIO PROCÓPIO
2015

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Cornélio Procópio
Programa de Pós-Graduação em Engenharia Mecânica

Título da Dissertação Nº 004:

"Desenvolvimento De Arquiteturas Híbridas Através De Sistemas Computacionais Inteligentes Aplicados À Robótica Autônoma".

por

Esdras Salgado Da Silva

Orientador: **Prof. Dr. Marcio Mendonça**

Esta dissertação foi apresentada como requisito parcial à obtenção do grau de **MESTRE EM ENGENHARIA MECÂNICA** – Área de Concentração: **Ciências Mecânicas,** linha de pesquisa: **Dinâmica De Sistemas Mecânicos**, pelo Programa de Pós-Graduação em Engenharia Mecânica – PPGEM – da Universidade Tecnológica Federal do Paraná – UTFPR – Câmpus Cornélio Procópio, às 14h do dia 14 de agosto de 2015. O trabalho foi aprovado pela Banca Examinadora, composta pelos professores:

Prof. Dr. Marcio Mendonça (Orientador – UTFPR-CP)	Prof. Dr. Edson Hideki Koroishi (UTFPR-CP)
Profa. Dra. Lúcia Valéria Ramos De Arruda (UTFPR-CT)	Prof. Dr. Ricardo Breganon (IFPR – Câmpus Jacarezinho)

Visto da coordenação:

Prof. Dr.Edson Hideki Koroishi
Coordenador do Programa de Pós-Graduação em Engenharia Mecânica
UTFPR Câmpus Cornélio Procópio

Agradecimentos

Agradeço a Deus pelo dom da vida e por estar comigo, fonte de minhas forças e direcionamento em todos os momentos desta jornada.

A minha família, pela motivação e apoio necessário desde os primeiros passos até essa importante etapa em minha vida, especialmente a meu pai Edson Veiga da Silva, minha mãe Ivone do Amaral Salgado, minha irmã Monique Hellen Salgado Chagas e meu segundo pai Marcos Antônio Cecílio Chagas.

A minha grande amiga Fernanda Tanaka por estar ao meu lado e pela força recebida, incentivando-me a esforçar e fazer o melhor de mim. A meu patrão e amigo Roberto C. Costa Takei (e família) pelo incentivo em relação aos meus estudos. A meu amigo Rafael Santos (in memoriam) que se tornou um irmão.

Ao Professor Orientador Márcio Mendonça por promover subsídio intelectual necessário para conclusão desta etapa, fazendo um trabalho excepcional, e, além disso, ser um ótimo amigo que incentivou meus planos. E á professora Lúcia Valéria Ramos de Arruda por todo apoio com revisões de artigos proporcionando condições para uma melhor escrita e pelo aceite do convite de participação da banca de defesa final.

A todos os professores do programa, em especial ao professor Adailton Borges pelo ótimo trabalho exercido, sendo, junto com o professor Marcio Mendonça, exemplo de excelência profissional e pessoal. E aos professores Ricardo Breganon e Edson Hideki Koroishi por aceitarem ser membros da banca de defesa final.

E por fim aos amigos mestrandos Bruno Shimada, Danilo Montilha, Lucas Niro e Wanderley Malaquias pelo apoio e amizade.

"So whether you eat or drink or whatever you do, do it all for the glory of God."
(1 Corinthians 10:31)

"Portanto, quer comais quer bebais, ou façais outra qualquer coisa, fazei tudo para glória de Deus."
(1 Coríntios 10:31)

RESUMO

SILVA, Esdras S. **Desenvolvimento de Arquiteturas Híbridas Através de Sistemas Computacionais Inteligentes Aplicados à Robótica Autônoma**. 2015. 117 f. Dissertação de Mestrado – Programa de Pós-Graduação em Engenharia Mecânica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2015.

Neste trabalho pretende-se apresentar as contribuições desenvolvidas, por meio de arquiteturas de controle de robôs (agentes) móveis baseadas em Lógica *Fuzzy* e Mapas Cognitivos *Fuzzy*, em especial, *Fuzzy* Ponderado Hierárquico, do inglês *Hierarquical Weighted Fuzzy* (HWF) e Mapas Cognitivos *Fuzzy* Híbridos Dinâmicos, do inglês *Hybrid Dynamic-Fuzzy Cognitive Maps* (HD-FCM). Essas arquiteturas serão aplicadas na geração de trajetória de um agente móvel, inicialmente em cenários virtuais e a *posteriori*, em uma plataforma robótica real. Serão comparadas as duas arquiteturas em simulações. Resultados iniciais apontam para a escolha do controlador HD-FCM para o controle do robô. Como proposta inicial o robô tem a funcionalidade de se deslocar entre dois pontos pré-estabelecidos. Os controladores estabelecem um trajeto viável podendo haver ou não obstáculos nos cenários. Simulações em ambientes virtuais são utilizadas para verificar a autonomia dos controladores desenvolvidos. Resultados, mesmo que ainda iniciais, com simulações em condições próximas das ideais, sugerem a viabilidade das propostas.

Palavras-chave: Agentes Autônomos. Sistemas *Fuzzy*. Mapas Cognitivos *Fuzzy*. Arquitetura Híbrida. Sistemas Computacionais Inteligentes.

ABSTRACT

SILVA, Esdras S. **Developing Architectures Hybrid Through Intelligent Computing Systems Applied to Autonomous Robotics**. 2015. 117 f. Masters Dissertation – Programa de Pós-Graduação em Engenharia Mecânica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2015.

This paper aims to present the main contributions developed through mobile robot control architectures (agents) based on Fuzzy Logic and Fuzzy Cognitive Maps, in particular, Hierarchical Weighted Fuzzy (HWF) and Hybrid Dynamic Fuzzy Cognitive Maps (HD-FCM). These architectures will be applied in generating trajectory of a mobile agent, initially in virtual scenarios and a posteriori, in a real robotic platform. The two architectures simulations are compared. Initial results point to the choice of HD-FCM driver to control the robot. As an initial proposal, the robot has the capability to travel between two preset points. Controllers establish a viable path and there may be no obstacles on scenarios. Simulations in virtual environments are used to verify the autonomy of the developed controllers. Results, although still early, with simulations in near ideal conditions, suggest the feasibility of the proposals.

Keywords: Autonomous Agent. *Fuzzy* Systems. *Fuzzy* Cognitive Maps. Hybrid Architecture. Intelligent Systems.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Pesquisas envolvendo robôs autônomos móveis têm ampliado perspectivas pela evolução tecnológica. Dentre essa evolução, destaca-se o desenvolvimento de técnicas computacionais inteligentes, como por exemplo, Lógica *Fuzzy*, Mapas Cognitivos *Fuzzy* (do inglês *Fuzzy Cognitive Maps* – FCM), Redes Neurais Artificiais, aplicados em diversas áreas de conhecimento, em especial, Agentes Autônomos (parte integrante do escopo deste trabalho), técnicas hibridas como sistemas Neuro-*Fuzzy*, ANFIS (*Adaptive Network-Based Fuzzy Inference System*) entre outras. Há uma gama de problemas desafiadores a serem resolvidos no que diz respeito à navegação autônoma móvel (SHAIKH *et al*., 2013). As dificuldades da robótica móvel estão relacionadas a diferentes domínios: percepção, localização, modelagem do ambiente e controle de tomada de decisões dinâmicas, entre outras (COSTA; GOUVÊA, 2010).

Neste contexto, além das dificuldades citadas, existem pesquisas envolvendo Robótica Móvel Autônoma, com maior nível de complexidade e podem ser aplicadas a diferentes áreas, como por exemplo, área de segurança (para evitar situações de risco ao ser humano), em explorações a regiões intransitáveis, (ou inviáveis), de difícil acesso e com alta periculosidade, e em robôs de serviços, constituindo área emergente de pesquisa e aplicação, conforme apresentado em Fracasso e Costa (2005), Henkel, Doerschuk e Mann (2009) e Doerschuk, Jiangjiang e Mann (2012).

Outra área de pesquisa em robótica é a de robôs exploradores que podem ser operados remotamente por humanos, entretanto, isso exige sistemas de comunicação, câmeras e operadores. De outro modo, sistemas computacionais inteligentes podem ser aplicados no desenvolvimento de robôs com capacidade de adaptação em diferentes cenários, utilizando leituras de sensores de distância, como por exemplo, sensores de ultrassom (MENDONÇA, 2011). Essas aplicações se tornam úteis em casos de exploração de ambientes naturais (inóspitos), nos quais se exige que o sistema tenha capacidade de tomadas de decisões autônomas (RUSSELL; NORVIG, 1995), (WOOLDRIDGE; JENNINGS, 1995), (CALVO; ROMERO, 2006), (BROGGI *et al*., 2008), (COSTA; GOUVÊA, 2010). A exploração

de ambientes traz dificuldades que devem ser tratadas, por exemplo, incertezas, imprecisões e erros sequenciais.

A complexidade envolvida nas tarefas de geração de trajetórias eficientes é reconhecidamente alta (MENDONÇA, 2011), devido a tomadas de decisões dinâmicas em sequência (uma tomada de decisão errada pode comprometer os passos seguintes, ocasionando problemas de localização e sensoriamento) (CALVO; ROMERO, 2006). Durante a exploração, nesse trabalho de dissertação entre outros, o agente necessita da habilidade de responder a eventos imprevisíveis, como obstáculos móveis e/ou inesperados (BAKAMBU, 2006), (MENDONÇA; ARRUDA; NEVES, 2011) para atingir um alvo considerando possíveis obstáculos no meio da trajetória.

Segundo Khodayari (2010), existem duas estratégias principais para desenvolvimento de agentes autônomos. O primeiro está relacionado a conhecimentos heurísticos, e é representado neste trabalho pela Lógica *Fuzzy e por* Mapas Cognitivos *Fuzzy*, conforme apresentado em Chiang *et al*. (2006), Yang e Zheng (2007), Cai, Rad e Chan (2010), e adaptado ao modelo utilizado. A segunda baseia-se na teoria de controle linear que não é o escopo dessa pesquisa.

1.1 OBJETIVO

Neste trabalho, de um modo geral, objetiva-se desenvolver arquiteturas de controle para um agente (robô) explorador autônomo com capacidade de tomadas de decisões dinâmicas sobre estratégias de navegação através da interação com o ambiente. De modo específico, o objetivo inicial desses controladores é fazer com que o robô atinja um ou mais alvos, com posições pré-determinadas e desvie de possíveis obstáculos no caminho. Por ser um robô com mais de um objetivo (atingir um ou mais alvos e desviar de obstáculos), esses controladores serão hierárquicos, necessitando assim arquiteturas hierárquicas a serem desenvolvidas. A proposta de desenvolvimento empregada é inspirada no trabalho de Braitenberg (1986), Coppin (2010), Mendonça et al (2011) a qual sugere a aplicação das técnicas computacionais inteligentes, inicializando-se a partir de um modelo simples com poucas ou uma funcionalidade, e gradativamente agregar novos objetivos a fim de

melhorar a capacidade de exploração do agente, como por exemplo, gerenciar a bateria e retornar a um ponto de recarga. Especificamente, neste trabalho o controlador inteligente em primeira instância terá a capacidade de deslocamento entre dois pontos previamente estabelecidos em um plano. O segundo objetivo será adicionar ao robô a funcionalidade de desvio de obstáculos presentes na trajetória inicialmente proposta. Outro objetivo, posteriormente incrementado, foi a localização de mais alvos. Neste caso existe uma memória de localização com a função para sua orientação em relação ao seu objetivo principal, permitindo assim continuar seguindo para o alvo. A princípio havia por objetivo apenas um alvo, depois a ideia se expandiu para mais alvos, motivando assim a aplicação em empresas, especificamente em transporte de peças nas dependências internas.

Observa-se que os desenvolvimentos dos controladores *Fuzzy* e FCM foram por meio de conhecimento heurístico, pela observação do comportamento dinâmico em tempo de simulações. Uma possível definição encontrada na literatura é a seguinte: “heurísticas são procedimentos que tratam problemas de otimização sem dispor das garantias teóricas nem de que a solução ótima exata seja obtida” (CUNHA et al, 2013). Essa definição é umas das premissas de sistemas computacionais inteligentes (MENDONCA, 2011).

Em resumo, o trabalho objetiva a criação de duas arquiteturas de controle, uma utilizando controlador *Fuzzy* e outra utilizando controlador FCM (*Fuzzy Cognitive Maps*) a serem comparadas e após uma delas escolhida para ser implementada no robô móvel. O desenvolvimento do robô também é parte componente do escopo deste trabalho, bem como a criação de um simulador que apresente as respostas matemáticas necessárias para a comparação entre as duas arquiteturas de controle.

1.2 MOTIVAÇÃO

A motivação desta dissertação está no desenvolvimento de métodos que possibilitem a criação de modelos baseados em Sistemas *Fuzzy* e/ou FCM, ou até mesmo novas versões baseadas nos modelos clássicos chamadas de HWF, do inglês *Hierarquical Weighted Fuzzy* e HD-FCM, do inglês *Hybrid Dynamic – Fuzzy*

Cognitive Maps a serem aplicados na realização de exploração a ambientes desconhecidos e inóspitos. Deste modo, serão empregadas neste trabalho duas técnicas de controle inteligente: Sistemas *Fuzzy*, e Mapas Cognitivos *Fuzzy*. Neste contexto, além dos Sistemas *Fuzzy* clássicos, optou-se por utilizar FCM (*Fuzzy Cognitive Maps*), cuja tradução nessa pesquisa foi convencionada por Mapas Cognitivos *Fuzzy*, os quais podem agregar características da coexistência de dados e conhecimento (PAPAGEORGIOU, 2014), como será apresentado ao longo dessa dissertação.

Após a simulação utilizando as duas arquiteturas, pretende-se fazer a comparação entre essas técnicas computacionais inteligentes apresentadas, visando listar seus aspectos positivos e negativos a fim de obter quais as vantagens e desvantagens para uma aplicação correta em diferentes ambientes de navegação, e após, implementar um dos controladores em um robô físico, com a possibilidade de início assim à possibilidade de futuros trabalhos, como tomadores de decisões para construção de robôs autônomos e, até possibilidades para sistemas de auxílio de tomadas de decisões como aplicações em cadeiras de rodas elétricas para uma maior comodidade e segurança de seres humanos. Outra área de possível investigação para essa pesquisa é a área de *Swarm Robotics* (robótica de enxame). Uma vez construída uma arquitetura baseada em conhecimento para um robô autônomo é possível, a posteriori, evoluir para controle de mais de um robô.

1.3 CONTRIBUIÇÕES

A principal contribuição do trabalho se dá no desenvolvimento de uma metodologia baseada em Sistemas *Fuzzy* Hierárquicos, com mais de uma base de regras e funcionalidades, o HWF (*Hierarquical Weighted Fuzzy*) e o uso de uma versão evoluída dos Mapas Cognitivos *Fuzzy* clássicos, o HD-FCM (*Hybrid Dynamic - Fuzzy Cognitive Maps*) para o desenvolvimento de sistemas computacionais inteligentes dinâmicos a partir de conhecimentos heurísticos e, possivelmente dados históricos, em especial com a aplicação destes métodos. Do ponto de vista do desenvolvimento de técnicas computacionais, a definição das diferenças, vantagens e desvantagens do Sistema *Fuzzy* comparado a FCM aplicados em um sistema

físico possibilitará uma melhor escolha da técnica a utilizar envolvendo aspectos como custo computacional, rapidez no processamento, viabilidade da técnica, entre outros. Por fim, uma terceira contribuição consiste em uma proposta de duas arquiteturas para sistemas inteligentes de duas camadas dinâmicas, que serão validadas pelo desenvolvimento de um sistema de navegação autônomo para robôs móveis.

Este sistema não se limita apenas às aplicações apresentadas, pode ser também aplicado na construção outros tipos de sistema ao qual seja necessário modelar a tomada de decisão *on-line* a partir de eventos dinâmicos.

As simulações feitas com incertezas (ruído branco da ordem de cinco por cento), conforme será apresentado no Capítulo 6 sugerem que as arquiteturas baseadas nas técnicas apresentadas, constituem ferramentas flexíveis e robustas, capaz de trabalharem com imprecisões e incertezas características dos sistemas dinâmicos reais.

1.4 ESTRUTURA DO TRABALHO

Esta dissertação encontra-se organizada da seguinte forma: o capítulo 2 compõe-se de fundamentos de controle *Fuzzy*. O capítulo 3 aborda especificamente o conceito de Mapas Cognitivos *Fuzzy* (FCM). O capítulo 4 apresenta a modelagem do robô. O capítulo 5 descreve os sistemas de navegação desenvolvidos, conceituando as arquiteturas utilizadas; O capítulo 6 apresenta os resultados simulados em seus diferentes cenários a fim de se sugerir autonomia. E, finalmente, a conclusão e futuros trabalhos serão apresentados no capítulo 7.

2 SISTEMAS DE CONTROLE *FUZZY*

Nesta seção serão apresentados os fundamentos da lógica *Fuzzy* e uma revisão bibliográfica sobre lógica *Fuzzy* aplicada à robótica móvel.

2.1 FUNDAMENTOS DE LÓGICA FUZZY

A inteligência computacional, segundo Zadeh (1968), é um conjunto de métodos baseados na teoria de conjuntos, termos linguísticos, lógica multivalorada, sistemas baseados em regras que visam trabalhar tolerâncias às falhas, imprecisões e incertezas de sistemas reais. Zadeh (1968) propõe o conceito de Lógica *Fuzzy*, com a fusão dessas teorias, que possibilita a modelagem de processos complexos (baseadas em informações imprecisas ou aproximadas), expressando regras de controle em termos linguísticos. Diversas áreas de conhecimento utilizam essa técnica como solução para aplicações de controle, reconhecimento de padrões e aproximação de funções, principalmente em casos de nenhum ou parcial conhecimento dos modelos matemáticos. Um exemplo clássico de modelagem *Fuzzy* é o desenvolvimento de um controlador através de conhecimento empírico e/ou heurístico por meio de termos linguísticos, funções de pertinência e uma ou mais bases de regras.

Lógica *Fuzzy*, ao contrário da lógica booleana (somente dois estados, "*on-off*") permite a modelagem de problemas de controle utilizando transições suaves entre os elementos considerados. De acordo com Gomide e Pedrycz (2007) "Conjuntos *Fuzzy* são funções de pertinência, que são mapeamentos de um dado universo de discurso para o intervalo unitário".

Um Sistema *Fuzzy* é composto por três principais componentes: entradas e saídas, conjunto de regras; e mecanismo de inferência (PASSINO, 1997). A Figura 1 apresenta a disposição desses componentes em um diagrama de Sistema *Fuzzy* genérico. Uma das dificuldades de se implementar a Lógica *Fuzzy* está no fato de aumentar o número de regras, de forma exponencial com o número de variáveis envolvidas, enquanto em uma estrutura hierárquica *Fuzzy*, o número total de regras

aumenta linearmente (WANG, 1999). Devem ser observadas também as inconsistências *Fuzzy*, como por exemplo, uma função de pertinência deve estar associada a pelo menos uma regra. Em Sistemas *Fuzzy* adaptativos, algoritmos retiram funções de pertinências velhas, ou seja, quando ficam sem ativação por um determinado período (DA SILVA, 2014).

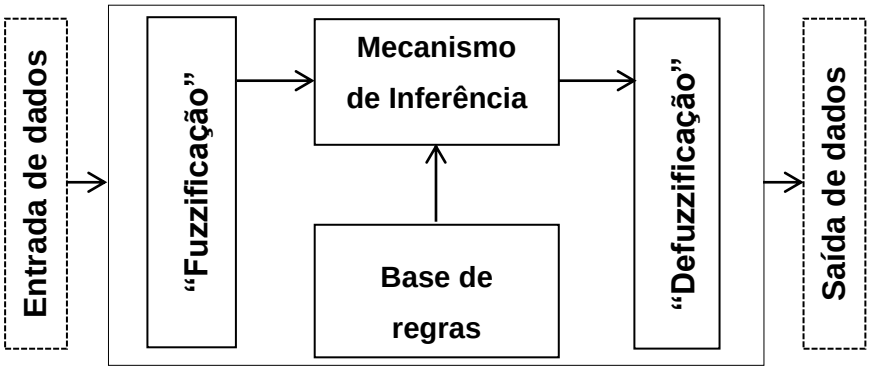


Figura 1 - Diagrama genérico da Lógica *Fuzzy*
Fonte: Adaptado de Passino (1997).

A estrutura de universo de discurso, o método de fuzzificação, o formato da base de regras, a máquina de inferência e o método de defuzzificação propostos por King e Mamdani (1977) formam um dos mais utilizados sistemas de inferência *Fuzzy* utilizados na literatura. Quando bem aplicado, este sistema de inferência pode contribuir para simplificação e aumento da velocidade de processamento e robustez do controlador, possibilitando decisões rápidas e coerentes num ambiente de incertezas, em situações críticas como, por exemplo, situações de falhas ou conflitos (FIGUEIREDO; TEIXEIRA, 1998). A Figura 2 ilustra em blocos o controlador lógico *Fuzzy* (*Fuzzy Logic Controller* – FLC) proposto por King e Mamdani (1977):

De acordo com a Figura 2 e, segundo Figueiredo *et al*. (1993), as etapas de projeto para construção de um controlador lógico *Fuzzy* consiste em:

1. Definir os universos de discurso das variáveis linguísticas do sistema através da observação do comportamento do sistema e/ou processo, como por exemplo, observando o erro e a variação do erro.

2. Definir o número de termos primários e graus de pertinência dos conjuntos *Fuzzy* que representam cada termo;

3. Determinar as regras que formam o algoritmo de controle que compõem o mecanismo de inferência ou motor de inferência; o mecanismo de

inferência emula as tomadas de decisões dos especialistas interpretando e aplicando o conhecimento modelado (PASSINO; YOURKOVIC, 1997).

4. Definir os parâmetros de projeto, como método de inferência, método de fuzzificação e defuzzificação e atuação do controlador.

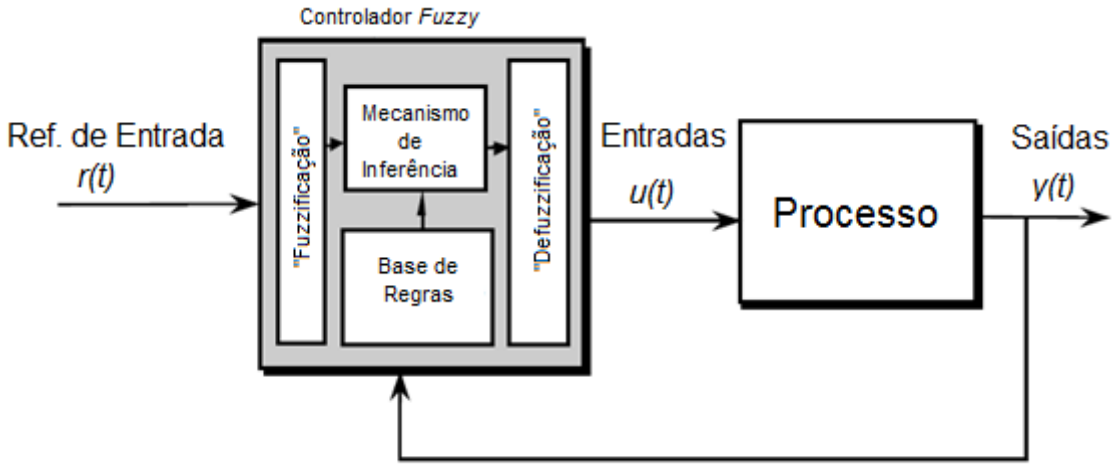


Figura 2 - Modelo do Controlador Lógico *Fuzzy*
Fonte: Adaptado de Passino (1998).

O propósito da utilização de um controlador FLC (*Fuzzy Logic Controller*) em um determinado processo é capturar o conhecimento de um operador e convertê-lo em um modelo para tomadas de decisões de controle. Esta conversão em controladores do tipo FLC utiliza variáveis linguísticas e regras. Além disso, estes controladores permitem modelar a incerteza dos processos, possibilitando desenvolver sistemas mais robustos, que podem ser utilizados em processos complexos (PASSINO; YOURKOVICH, 1997). Um controle do tipo FLC para operar no chão de fábrica deve ter as seguintes funcionalidades:

- Controlar e operar automaticamente processos complexos não lineares multivariáveis com desempenho pelo menos equivalente aos operadores.
- Respeitar as especificações e restrições operacionais.
- Ser robusto e operar em tempo real (PADILHA, 2001).

O controle *Fuzzy* tem se mostrado também uma alternativa viável ao controle clássico para processos com parâmetros variantes no tempo, não lineares e com informações imprecisas. Para esses casos, a aplicação de um Controlador *Fuzzy* frequentemente oferece um desempenho igual ou superior aos controladores adaptativos, do tipo preditivo ou por modelo de referência (CALLAI *et al*., 2007).

Para o controle supervisório, os sistemas baseados em Lógica *Fuzzy* podem fornecer uma solução eficiente através do desenvolvimento de controladores com base na experiência de especialistas e/ou operadores em vez de utilizar modelos matemáticos complexos, como se utilizam nas abordagens baseadas em controle clássico, robusto entre outros. Uma das tarefas do supervisor é a de, por exemplo, gerar *set-points* para os controladores nos níveis inferiores da planta ou ainda gerenciar estratégias de recuperação da campanha em presença de perturbações. Essas atividades podem ser implementadas através de supervisores *Fuzzy* (ARRIAGA-DE-VALLE; DIECK-ASSAD, 2006).

A Lógica *Fuzzy* pode apresentar diversas vantagens e desvantagens, das quais as principais serão listadas a seguir:

Vantagens:

- Por ser uma abordagem intuitiva, torna o entendimento relativamente fácil;
- Tolerância a dados imprecisos e flexibilidade;
- Possibilidade de modelar funções não lineares de complexidade arbitrária.
- Possibilidade de criar um Sistema *Fuzzy* para combinar com qualquer conjunto de dados de entrada e saída. Através de técnicas adaptativas como *Adaptive Neuro-Fuzzy Inference Systems* (ANFIS);
- Possibilidade de ser construída com base na experiência de especialistas;
- Em contraste direto com Redes Neurais, que levam dados de treinamento e gerar modelos opacos, impenetráveis, a Lógica *Fuzzy* permite contar com a experiência de pessoas que já entendem o sistema;
- Pode ser usada em conjunto com técnicas de controle convencionais;
- A linguagem natural, que é usada por pessoas comuns em uma base diária, tem sido formada por milhares de anos de história humana a ser conveniente e eficiente. As frases escritas na linguagem comum representam um triunfo da comunicação eficiente, e são empregadas nos Sistemas *Fuzzy* através da base de regras.

Desvantagens:

Com referência às desvantagens da Lógica *Fuzzy*, as seguintes questões são motivo de preocupação ao empregar uma estratégia de reunir conhecimento controle heurístico (PASSINO; YOURKOVICH, 1997):

• Será que os comportamentos que são observados por um especialista humano e utilizados para construir o Controlador *Fuzzy* incluem todas as situações que podem ocorrer devido a distúrbios, o ruído ou variações dos parâmetros da planta?

• Pode o especialista humano de forma realista e confiável prever problemas que podem surgir com instabilidades do sistema de circuito fechado ou ciclos-limite?

• Será que o especialista humano é capaz de incorporar efetivamente critérios de estabilidade e objetivos de desempenho (por exemplo, tempo de estabilização, "*overshoot"*, e as especificações de rastreamento) em uma base de regras para garantir que a operação de confiança pode ser obtida?

Essas questões podem parecer ainda mais complexas se o problema de controle envolve um ambiente de segurança crítica em que a falha do sistema de controle para atender os objetivos de desempenho pode levar a perda de vidas humanas ou um desastre ambiental, ou se o conhecimento do especialista humano implementado no Controlador *Fuzzy* é um pouco inferior ao do especialista mais experiente que deveria se esperar para a concepção do sistema de controle (designers diferentes têm diferentes níveis de perícia). É evidente que, em seguida, para algumas aplicações, há uma necessidade de desenvolver uma metodologia para, implementar e avaliar Controladores *Fuzzy* para garantir que eles são confiáveis no cumprimento de suas especificações de desempenho (PASSINO; YOURKOVICH, 1997). Por serem problemas envolvendo a capacidade do ser humano em quantificar de forma qualitativa os problemas, essas desvantagens ainda permeiam a Lógica *Fuzzy* até a presente data.

Nessa dissertação, um sistema de navegação robótico baseado em FLC será usado para que o agente atinja os objetivos de chegar ao alvo e desviar de obstáculos na subseção 5.3.

2.2 REVISÃO BIBLIOGRÁFICA SOBRE APLICAÇÕES DE LÓGICA *FUZZY* EM ROBÓTICA MÓVEL

As primeiras noções da lógica dos conceitos “vagos” foram desenvolvidas pelo lógico polonês Jan Lukasiewicz (1878-1956) em 1920 que introduziu conjuntos com graus de pertinência sendo 0, ½ e 1 e, mais tarde, expandiu para um número infinito de valores entre 0 e 1, o que precedeu a Lógica *Fuzzy*.

O primeiro a introduzir o conceito de Lógica *Fuzzy* foi Lotfi Asker Zadeh (Professor da Universidade da Califórnia, Berkley), que ao observar que os recursos tecnológicos disponíveis eram incapazes de automatizar as atividades de diversas naturezas tais como industrial, química ou biológica que compreendessem situações ambíguas impossíveis de se implementar utilizando lógica computacional fundamentada na lógica booleana, publicou um artigo em 1965 (Zadeh, 1965) combinando os conceitos da lógica clássica e os conjuntos de Lukasiewicz, revolucionando o conceito de lógica até então, criando a Lógica *Fuzzy*.

A partir daí iniciou-se uma série de estudos utilizando a aplicação de lógica *Fuzzy* nas mais diversas áreas principalmente na indústria europeia e japonesa nas décadas de 70 e 80. Em nossa revisão literária nos ateremos em apresentar alguns trabalhos importantes voltados para a área de robótica móvel.

Em 1995, o francês Praxitele (Garnier *et al*, 1995) controlou uma frota de veículos elétricos para serem utilizados pela população no lugar de transportes coletivos com o intuito de evitar estacionamentos lotados. Este projeto utilizou técinicas de lógica *Fuzzy* e linhas de fugas em um controle híbrido.

Em Franz *et al* (1998) foi desenvolvido um sistema para navegação e guiagem de robôs móveis autônomos em que a informação de posição e atitude do veículo é utilizada por um sistema de guiagem, baseado em controlador tipo *Fuzzy*, que é de sintonização simples e prescinde do conhecimento do modelo dinâmico do veículo. O sistema completo é implementado em tempo real.

Ainda em 1998, Sandi *et al*. (1998) desenvolveu sistema para guiagem de robôs móveis em que um controlador *Fuzzy* utiliza a informação de posição e latitude do veículo para o sistema de guiagem, de sintonização simples e prescinde do conhecimento do modelo dinâmico do veículo. O sistema completo é implementado em tempo real, apresentando desempenho considerável. Dois modos de

visualização do controle dos robôs caracterizam esse sistema; um a navegação que designa a determinação de posição e orientação do veículo em um dado instante de tempo e outro a guiagem, que se refere ao controle da trajetória.

Ramirez *et al* (2007) descreve um sistema de navegação autônoma para um veículo autônomo utilizando técnicas de visão de máquina aplicados a imagens em tempo real capturadas durante o trajeto, para fins académicos. A experiência consiste na navegação autônoma de um carro de controle remoto através de um circuito fechado. Técnicas de visão computacional são utilizadas para a detecção de ocorrências no meio ambiente. As imagens recebidas são capturadas no computador através do cartão de aquisição NI USB-6009, e processadas em um sistema desenvolvido sob a plataforma LabVIEW, aproveitando o conjunto de ferramentas para aquisição e processamento de imagem. Técnicas de controle de lógica *Fuzzy* são incorporadas para as decisões de controle intermédios necessários durante a navegação do carro. Uma abordagem eficiente com base na lógica de máquina-estados é utilizada como um método ideal para implementar as mudanças exigidas pelo controle de lógica *Fuzzy*.

Farhi (2008) descreve o modelo do sistema de controle *Fuzzy* para um carro robô autônomo que atua e um ambiente desconhecido, imprevisível e dinâmico. O sistema de controle *Fuzzy* fornece a fusão de dados de vários sensores e tem a função de garantir uma navegação do robô autônomo. Duas estratégias são utilizadas. Uma de controle de desvio de obstáculos e outra de controle de seguimento do alvo. Eles são combinados e, assim, resultam em uma arquitetura de controle otimizada com sistema de controle *Fuzzy*.

Em Llorca *et al* (2011) tem por âmbito de estudo um sistema anti-colisão (CAS) para veículos autônomos, com foco na prevenção de colisões de pedestres. O componente de detecção envolve um sistema de sensoriamento de pedestres baseado em estéreo de visão que fornece medições adequadas da hora de colisão. A manobra de prevenção de colisões é realizada utilizando controladores *Fuzzy* para os atuadores que imitam o comportamento humano, juntamente com uma alta precisão Sistema de Posicionamento Global (GPS), que fornece as informações necessárias para a navegação autônoma. Os testes de campo realizados apresentaram resultados encorajadores e provou a viabilidade da abordagem proposta.

Em Nogueira (2013) é apresentado o desenvolvimento de um sistema de navegação de um robô móvel autônomo utilizando-se de técnicas da lógica *Fuzzy*, desenvolvida na linguagem de descrição de hardware, VHDL, e a sua implementação em uma placa DE2 do fabricante Altera®. O objetivo do sistema de navegação proposto é o de que o protótipo do robô caminhe sem colidir em nenhum obstáculo utilizando para comunicação externa com o ambiente, três sensores de distância localizados um em cada lado do robô e um na frente, atuando através dessas informações em dois motores de passo. Os resultados desta implementação obtidos em simulação foram satisfatórios e comparados aos resultados obtidos como o mesmo processo no MATLAB.

Estes foram alguns dos trabalhos relevantes na área de controle *Fuzzy* voltados para a robótica móvel aos quais serviram de inspiração e material de estudo para esta dissertação.

3 MAPAS COGNITIVOS *FUZZY* (FCM)

Serão apresentados nesta seção os fundamentos de Mapas Cognitivos *Fuzzy* (FCM) e uma revisão bibliográfica sobre FCM aplicado à robótica móvel.

3.1 FUNDAMENTOS DE FCM

Para apresentar os conceitos necessários para a fundamentação do FCM de forma mais clara, esta seção foi dividida em duas partes. A primeira parte (subseção 3.1.1) discorre sobre os conceitos clássicos e básicos desta ferramenta. A segunda parte (subseção 3.1.2) apresenta as evoluções do FCM e suas vantagens e desvantagens.

3.1.1 FCM Clássico

Os Mapas Cognitivos foram inicialmente propostos por Axelrod (1976) para representar palavras, ideias, tarefas ou outros itens ligados a um conceito central e dispostos radialmente em volta deste conceito. São diagramas que representam conexões entre porções de informação sobre um tema ou tarefa. Os elementos são arranjados intuitivamente de acordo com a importância dos conceitos, e são organizados em grupos, ramificações ou áreas.

Axelrod (1976) desenvolveu também um tratamento matemático por meio de operações com matrizes e baseado na teoria de grafos, para seus Mapas Cognitivos, nos quais crenças ou afirmações a respeito de um domínio de conhecimento limitado são expressas por meio de palavras ou expressões linguísticas, interligadas por relações simples de causa e efeito (causa / não causa).

Estes mapas podem ser considerados um modelo matemático da “estrutura de crenças” de uma pessoa ou grupo, que permitem inferir ou predizer as consequências que esta organização de ideias causa no universo representado.

Este modelo matemático foi adaptado para inclusão de incertezas através da Lógica *Fuzzy* por Kosko (1986), gerando os Mapas Cognitivos *Fuzzy*. À semelhança dos Mapas Cognitivos originais, os FCM são grafos direcionais (dígrafos), em que os valores numéricos são variáveis ou conjuntos *Fuzzy*. Os "nós" destes grafos são conceitos linguísticos, representados por conjuntos *Fuzzy* e cada "nó" é associado a outros através de conexões. A cada uma dessas conexões é associado um peso numérico, que representa uma variável *Fuzzy* relacionada com o nível de causalidade entre os conceitos, como mostra a Figura 3.

Esses mapas conceituais, *Fuzzy* ou clássicos, representam o conhecimento individual ou de um grupo de especialistas, através dos valores de seus conceitos e dos pesos das suas relações. O conhecimento explícito é representado através das atribuições de valores a conceitos e relações causais, e o conhecimento implícito é representado através da estrutura do mapa. No entanto, permanece a dificuldade de transformar o conhecimento em modelos de comportamento humano. Na construção dos mapas conceituais esta dificuldade se reflete principalmente na definição da semântica e do relacionamento entre os conceitos (PACHECO, 2005). Além desta dificuldade de construção, a aquisição de dados deve ser bastante criteriosa, no sentido de se transformar dados em informação consistente para o mapa cognitivo.

Os Mapas Cognitivos *Fuzzy* usam uma representação simbólica pra descrever e modelar um sistema, o qual utiliza conceitos pra ilustrar diferentes comportamentos dinâmicos integrando a experiência de especialistas na construção do modelo. Como citado anteriormente, os FCM são grafos direcionados acíclicos em que os "nós" são conceitos linguísticos, representados por conjuntos *Fuzzy*, e cada "nó" é associado com outros através de "arcos". A cada um destes "arcos" é associado um peso numérico, que representa uma variável *Fuzzy* relacionada com o nível de causalidade entre os conceitos. Cada conceito representa uma característica do sistema, em geral: eventos, ações, metas, valores e tendências do sistema modelado. Além disso, cada conceito é caracterizado por um número que representa seu valor e é o resultado da normalização, em um intervalo [-1,1], do real valor da variável do sistema para qual este conceito foi modelado. Este intervalo representa a incerteza na relação causal que leva de um nó a outro.

De um modo geral pode definir que o FCM é uma metodologia de causalidade baseada no conhecimento para modelar sistemas complexos de decisão, originado a partir da combinação de Lógica *Fuzzy* e Redes Neurais

(KOSKO, 1986). Um FCM descreve o comportamento de um sistema desconhecido em termos de conceitos, cada conceito representa uma entidade, um estado, uma variável, ou uma característica do sistema (KOSKO, 1992).

Um exemplo de FCM é mostrado na Figura 3, o grafo da figura identifica as respectivas conexões causais, representadas pelos sinais e termos linguísticos, e os conceitos do mapa. Na prática, esses termos linguísticos são traduzidos em valores numéricos por meio de números *Fuzzy*, por exemplo, por média de avaliação dos especialistas, ou até mesmo média olímpica dos especialistas (se desconsidera o maior e menor valor sugerido no cálculo da média).

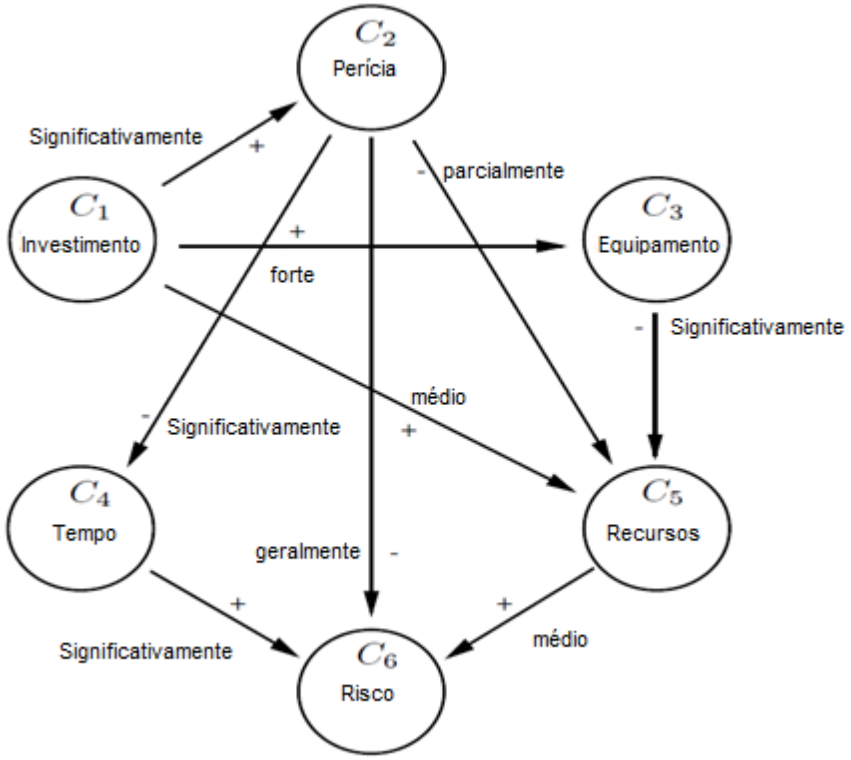


Figura 3 – Exemplo de um FCM (grafo)

Fonte: Adapt. de http://www.ibimapublishing.com/journals/CIBIMA/.html 2011.

Os conceitos de um mapa cognitivo (C_1 a C_6) podem ser atualizados através da iteração com os outros conceitos por meio das interconexões (relações causais $w_{i,j}$) e com o seu próprio valor. Isto é dado pela matriz (1) com os pesos das relações causais, e são representadas pelo peso da somatória.

$$w(i,j) = \begin{bmatrix} 0 & w_{12} & 0 & 0 & w_{15} \\ 0 & 0 & w_{23} & 0 & w_{25} \\ 0 & w_{32} & 0 & w_{34} & 0 \\ w_{41} & 0 & w_{43} & 0 & w_{45} \\ w_{51} & 0 & 0 & w_{54} & 0 \end{bmatrix} \quad (1)$$

Os valores dos conceitos vão evoluindo após várias iterações, como mostra a equação (2) até normalmente estabilizarem-se num ponto fixo ou num ciclo limite (STYLIOS; GROUMPOS, 1998).

$$A_i = f\left(\sum_{\substack{j=1 \\ j \neq i}}^{n} (A_j \times W_{ji}) + A_i^{anterior}\right) \quad (2)$$

Onde ***j*** é o contador das iterações, ***n*** é o número de nós no grafo, $\boldsymbol{W_{ji}}$ é o peso do arco que conecta o conceito $\boldsymbol{C_j}$ ao conceito $\boldsymbol{C_i}$, $\mathbf{A_i}$ ($\mathbf{A_i}^{\text{anterior}}$) é o valor do conceito $\boldsymbol{C_i}$ na iteração atual (anterior) e a função ***f*** (equação 3) é uma função do tipo sigmoide:

$$f(x) = \frac{1}{1 + e^{-\lambda x}} \quad (3)$$

Em alguns casos, o FCM pode não estabilizar e oscilar, ou até mesmo apresentar um comportamento caótico (STYLIOS; GEORGOPOULOS; GROUMPOS, 1999), Mendonça (2011). Geralmente, para sistemas bem-comportados, observa-se que após um número finito de iterações, os valores dos conceitos atingem um ponto de equilíbrio fixo ou um ciclo limite. Existem algumas variações da função de aprendizado utilizada (GLYKAS, 2010) entre métodos de aprendizagem.

Esta interatividade entre conceitos permite a construção de mapas de forma modular, em que pequenos mapas são criados para representar partes do problema e depois interconectados entre si. Esses pequenos mapas são testados de forma independente e só depois estas partes são conectadas para compor um mapa

maior. A interatividade entre Mapas Cognitivos é certamente uma vantagem na construção de modelos de um sistema mais complexo, devido a possibilidade de se construir módulos diferentes de forma independente e conectá-los, *a posteriori* (MIAO, *et al*; 2001).

Como demonstrado em Bose e Liang (1996); Silva, Spatti e Flauzino (2010) uma RNA (Redes Neurais Artificiais) com conexões diretas (*feed forward*) é equivalente a um grafo direto acíclico. Nesse contexto, pode-se utilizar na construção e interpretação de um FCM como variação de Redes Neurais Artificiais, com um nó ou conceito corresponde a um neurônio; os arcos ou relações causais entre conceitos podem ser obtidos via treinamento ou de forma manual, a partir dos dados históricos do sistema ou pelo conhecimento do (s) especialista (s), respectivamente.

3.1.2 Evoluções do FCM clássico

Trabalhos da literatura consideram o FCM como uma evolução dos Mapas Cognitivos (MIAO *et al*, 2001, 2010), (STYLIOS *et al*, 1998, 1999, 1999b), (DICKERSON; KOSKO, 1994, 1996) e, portanto, de um grafo, em que foram agregadas características semânticas dos Sistemas *Fuzzy* e propriedades de plasticidade e estabilidade das Redes Neurais (STYLIOS *et al*., 2008).

Esta capacidade de auto-organização, ou plasticidade, contribui para flexibilidade no desenvolvimento e aplicações de modelos computacionais inteligentes através de FCM. No entanto, apesar da semelhança estrutural com as Redes Neurais Artificiais, os FCM são inicialmente construídos a partir de conhecimento empírico em que cada conceito tem um significado físico e cada conexão representa uma relação de causa-efeito entre os mesmos. Os valores iniciais dos conceitos e relações causais também têm correspondência com o comportamento físico do sistema modelado. Esta característica é inexistente nas Redes Neurais Artificiais e tem sido um forte argumento contra a interpretabilidade deste tipo de modelo (HAYKIN, 2000).

Em resumo, FCM é uma opção de modelagem para representar sistemas complexos que não estão bem definidos, combinando as potencialidades das

abordagens cognitivas de abstração de conhecimento por meio de termos linguísticos dos Sistemas *Fuzzy* às vantagens de modelagem oriundas das Redes Neurais Artificiais, devido à sua estrutura em grafos, capacidade de aprendizagem, entre outras. De forma resumida na modelagem inicial os FCM têm aspectos da Lógica *Fuzzy*, das Redes Semânticas. Entretanto, permite a inclusão de dados históricos para refinamento do modelo, através de métodos heurísticos, Algoritmos Genéticos, *Particle Swarm* entre outras técnicas (STYLIOS *et al*., 2008). E como foi aplicado nesse trabalho algoritmos de sintonia dinâmica como aprendizado de Hebb, oriundos originalmente das Redes Neurais Artificiais, e permite também a inclusão de algoritmos de aprendizagem por reforço, como por exemplo em Mendonça (2011).

Essa técnica apresenta vantagens e desvantagens, das quais serão apresentadas a seguir (CHUNMEI, 2008).

Vantagens:

- É uma ferramenta de computação suave dado pelo resultado da sinergia da Lógica *Fuzzy* e metodologias de Redes Neurais Artificiais.
- Apresenta um método interativo de aquisição de conhecimentos ou aprendizado usando método de construção intuitiva e heurística.
- Oferece um quadro mais flexível e poderoso para representar o conhecimento humano e para o raciocínio, ao contrário de sistemas especialistas tradicionais que explicitamente implementam regras com "*If / Then*".
- Enfatiza as conexões de conceitos como unidades básicas para armazenar conhecimento, e a estrutura representa o significado de sistema.
- Pode ajudar a descrever a estrutura esquemática, representando as relações causais entre os elementos de um determinado ambiente de decisão, e a inferência pode ser calculada por operação matriz numérica.
- É constituído por poucas linhas de cálculo, o que permite implementar em um controlador simples, ou seja, de baixo custo.
- Relações causais entre os nós são difusos. Em vez de usar apenas sinais que indicam causalidade positivo ou negativo, utiliza um número

associado com a relação para expressar o grau de relação entre dois conceitos.

- O sistema é dinâmico, envolvendo realimentação, em que o efeito da mudança no nó conceito afeta outros nós, influenciando nos dados de saída. A presença dessa realimentação adiciona um aspecto temporal para a operação do FCM.

Desvantagem:

- Apesar do passo de sintonia, modelos de inferência baseados em FCM não apresentam robustez na presença de modificações dinâmicas modeladas a partir desta representação do conhecimento “rígido” por meio de operações com grafos e matrizes (ACAMPORA; LOIA, 2011).

Para contornar esse problema, este trabalho desenvolve um novo tipo de FCM em que os conceitos e as relações causais são dinâmicos no gráfico a partir da ocorrência de eventos. Deste modo, o modelo de mapa cognitivo distorcido dinâmico é capaz de adquirir e usar o conhecimento heurístico dinamicamente.

O presente trabalho explora as potencialidades dos FCM’s a fim de desenvolver uma nova arquitetura para modelagem de um sistema dinâmico. A proposta de HD-FCM e sua aplicação na navegação autônoma serão apresentadas na Seção 5.4.

3.2 REVISÃO BIBLIOGRÁFICA SOBRE FCM COM ENFOQUE EM ROBÓTICA MÓVEL

Os Mapas Cognitivos *Fuzzy* (FCM) surgiram baseados na formalização de Axelrod (1976) que introduziu formalmente o conceito de mapas cognitivos como forma de representação do conhecimento e modelagem de decisões. Dez anos após essa representação formal, Kosko (1986) ao estudar esses Mapas Cognitivos, incluiu a esses conceitos a lógica *Fuzzy*, originando assim os Mapas Cognitivos *Fuzzy* (FCM).

A partir daí, iniciou-se uma série de estudos e aplicações desse sistema em diferentes áreas de conhecimento tais como vida artificial (DICKERSON; KOSKO, 1994, 1996), sistemas sociais (TABER, 1994), sistemas de tomada de decisão em rodovias de acesso rápido (PERUSICH, 1996), modelagem e tomada de decisão em ambientes corporativos e comércio eletrônico (LEE; LEE, 2003), visão robótica, mais especificamente em detecção de pontos em imagens geradas por sistema de câmeras estéreo (PAJARES; DE LA CRUZ, 2006), tomada decisão na área médica (PAPAGEORGIOU; STYLIOS; GROUMPOS, 2007), entre outros. Focaremos nossa revisão bibliográfica nos trabalhos voltados para robótica autônoma.

Foi desenvolvido em Golmohammadi (2006), um modelo a ser utilizado para a seleção de ação em robôs que se destinam a imitar o processo de raciocínio do ser humano em ambiente de simulação. Este modelo é construído em uma FCM e um algoritmo de aprendizagem de Hebb não linear. O modelo foi testado através de uma série de experiências práticas com uma versão do servidor de simulação de futebol ambiente 3D. Os fatores envolvidos foram definidos cuidadosamente para medir o desempenho da equipe. Os resultados obtidos mostraram uma melhora significativa no desempenho geral.

Buche, (2010) foca na simulação de comportamento de agentes autônomos em ambientes virtuais. O comportamento desses agentes deve determinar suas respostas não só aos estímulos externos, mas também em relação a estados internos. Para descrever tal comportamento foi proposto Mapas Cognitivos *Fuzzy* (FCM), segundo o qual estes estados internos podem ser explicitamente representados. O artigo apresenta o uso o FCM como ferramenta para especificar e controlar o comportamento dos agentes individuais. Em primeiro lugar, o FCM foi utilizado para modelar o comportamento. Em seguida, um algoritmo de aprendizagem é utilizado permitindo a adaptação das FCM's através da observação.

Em Mendonça *et al* (2011) foi desenvolvido um sistema de navegação autônoma utilizando mapas cognitivos *Fuzzy* (FCM). Uma nova variante da FCM, chamado *Dynamic – Fuzzy Cognitive Maps* (D-FCM), é proposto para modelar tarefas de decisão e / ou fazer inferências em navegação autônoma. Arcos da FCM são atualizados a partir da ocorrência de eventos especiais como a detecção de obstáculos dinâmicos. Como resultado, o modelo desenvolvido é capaz de representar o comportamento dinâmico do robô na presença de alterações de ambiente. Esta habilidade foi alcançada através da adaptação das relações entre

conceitos FCM. Um algoritmo de aprendizagem reforço também foi utilizado para ajustar o comportamento do robô. Alguns resultados da simulação são discutidos com destaque para a capacidade do robô autônomo para navegar entre obstáculos (navegação em ambiente desconhecido). Um sistema de navegação baseado em *Fuzzy* é usado como uma referência para avaliar o desempenho do sistema de navegação autónoma proposto. Os resultados foram satisfatórios em ambiente de simulação. Um comparativo com obras de lógica *Fuzzy* e futuros também são abordados.

Ganganath *et al* (2013) propõe um mapa cognitivo *Fuzzy* (FCM) com base na avaliação da situação (SAF - *situation assessment framework*) para a detecção de um alvo para um robô móvel. Um determinado objetivo de navegação é descrito com várias submetas utilizando conhecimento prévio. Com base nessas submetas a SAF proposta opera de forma recursiva sobre a meta de navegação para verificar e selecionar essas submetas. O sistema de tomada de decisão combina informação sensorial a partir de múltiplos sensores para verificar as submetas. A FCM é usada como um motor de raciocínio alto nível. Resultados experimentais demonstram que o quadro proposto pode decidir com precisão as metas de navegação em ambientes desconhecidos com base na informação sensorial e conhecimento especializado (GANGANATH *et al* 2013).

Motlagh (2012) utilizou um mecanismo de inferência causal baseado em mapas cognitivos *Fuzzy* (FCM) para controlar um robô móvel resolvendo problemas de localização levando em conta erros sequenciais e derrapagem das rodas. O trabalho busca apresentar também as vantagens do sistema FCM em relação às outras técnicas baseadas em regras. O sistema desenvolvido foi implementado em uma plataforma Pioneer. Os resultados e as comparações com os trabalhos relacionados são dados usando simulação *ActivMedia* e uma ferramenta de simulação desenvolvida FCM. Uma técnica de estimativa de erro é usada para medir o erro entre o real e os resultados da simulação.

Nota-se que a aplicação de FCM em robótica autônoma é um assunto que vêm sendo discutido há cerca de 10 anos, e notam-se grandes avanços, com resultados consideráveis para a aplicação em robótica autônoma.

4 MODELAGEM DO ROBÔ

Existem duas maneiras de representação dos robôs de acordo com o objetivo da aplicação: cinemática ou dinâmica. Os modelos cinemáticos são usados para representar os robôs, neste trabalho denominados de agentes, em função da velocidade e orientação das rodas. Os modelos dinâmicos tentam representar as forças generalizadas aplicadas pelos atuadores. Há dois modelos cinemáticos adotados para representar agentes: modelo de postura e modelo de configuração. O primeiro considera como estado apenas a posição e a orientação do robô, enquanto o segundo considera além da postura outras variáveis internas, como por exemplo o deslocamento angular das rodas. O modelo de postura satisfaz as necessidades do ponto de vista de controle de posição e orientação espacial do robô. A maioria dos trabalhos apresentados na literatura descreve o robô em função de coordenadas cartesianas utilizando apenas um modelo cinemático de postura (BLOCH; REYHANOGLU; MCCLAMROCH, 1992), (KOLMANOVSKY; MCCLAMROCH, 1995), (MURATA; HIROSE, 1993).

4.1 MODELAGEM CINEMÁTICA

O robô utilizado como modelo neste trabalho possui duas rodas móveis (localizadas na parte frontal e posterior do carro) e duas rodas fixas nas laterais (ligadas a um motor cada uma); três sensores ultrassônicos posicionados estrategicamente na parte frontal para captar possíveis obstáculos localizados na rota a ser seguida pelo robô.

Uma foto deste robô é apresentada na Figura 4, e seu modelo esquemático pode ser observado na Figura 5.

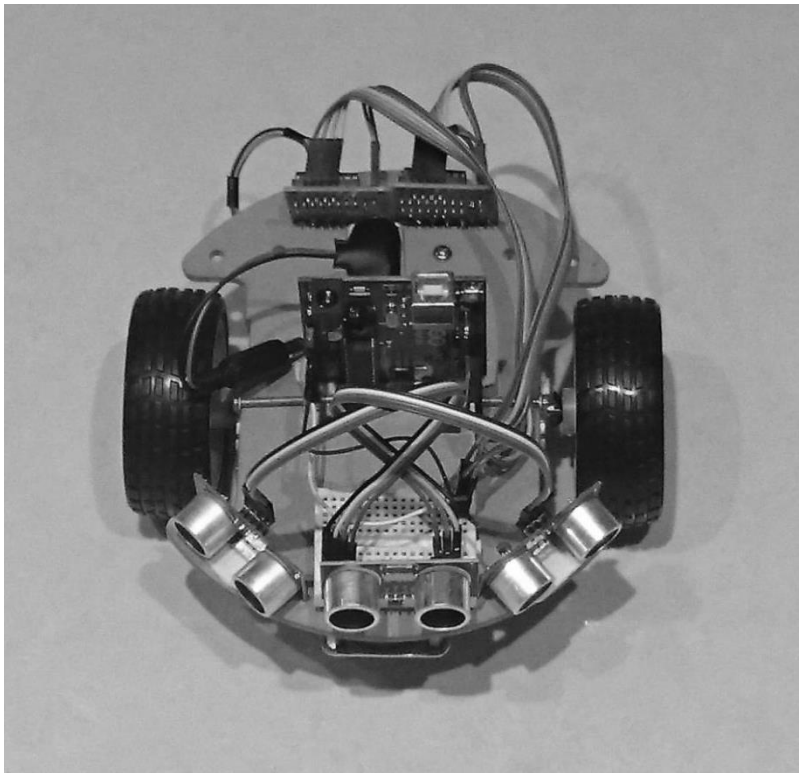


Figura 4 – Robô utilizado neste trabalho

Fonte: Autoria própria.

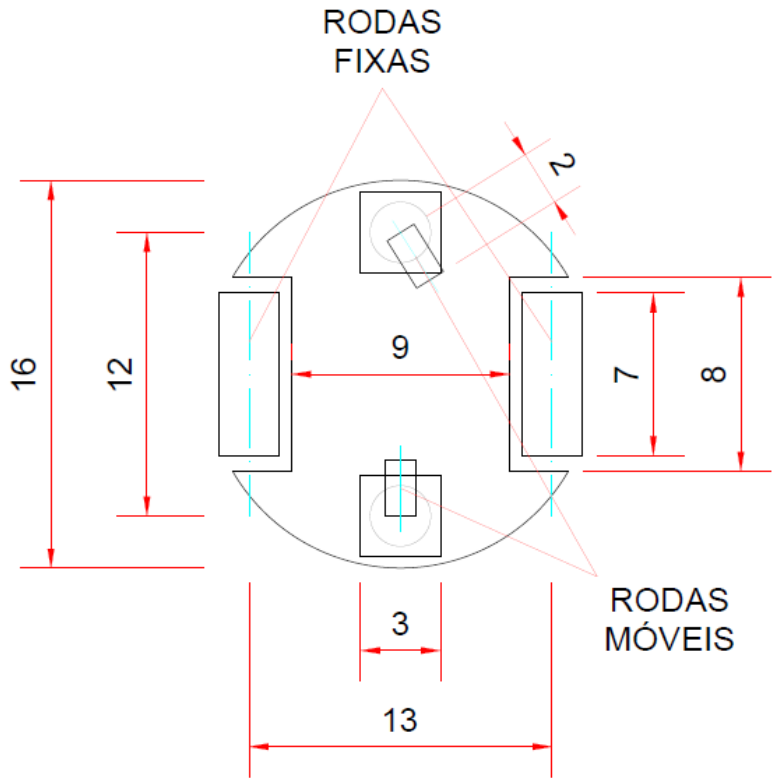


Figura 5 - Modelo esquemático do robô da figura 4

Fonte: Autoria própria.

4.2 MODELAGEM SIMULADOR

Um simulador em duas dimensões foi implementado no MatLab (versão 2012) para observar o comportamento dinâmico do agente móvel. As dimensões do protótipo e do cenário estão em uma escala de 1:100.

O trabalho de Russell e Norvig (1995) sugere que para o agente ser considerado autônomo, é necessário que se tenha sucesso em ao menos três simulações diferentes. Portanto a disposição de obstáculos, posição inicial do agente móvel e o ponto alvo são ajustáveis no programa, possibilitando a construção de diferentes cenários. No simulador desenvolvido, os sensores foram programados para que o envio de sinais ocorra a uma distância máxima de 15 cm, para que o agente não permaneça desviando de obstáculos que se encontram a uma distância segura do mesmo.

No presente capítulo serão apresentados todos os cálculos utilizados no processo, envolvendo desde o simulador, até o código do *MatLab*. Poder-se-ia utilizar o modelo clássico de translação e rotação (CRAIG, 1989). Entretanto, devido a divisão do espaço de busca em quadrantes e implementação do programa no Arduino (10.5.2), optou-se por um modelo baseado em funções geométricas. O modelo matemático utilizado e o procedimento do algoritmo utilizado são apresentados nas subseções 4.2.1 e 4.2.2, respectivamente.

4.2.1 Modelo Matemático Utilizado no Simulador

Nas subseções a seguir serão apresentados os procedimentos utilizados para a formulação do simulador. A Subseção 4.2.1.1 apresenta o procedimento adotado para que o robô se localize, quantificando sua distância em relação ao alvo a cada passo. A subseção 4.2.1.2 demonstra o procedimento para a transformação dos pulsos em movimento

4.2.1.1 Cálculo de distâncias

Ao adotar um plano cartesiano e localizar nesse plano a posição do móvel e do alvo, torna-se trivial obter as distâncias “x” e “y” entre esses dois elementos. Entretanto para a implementação do controlador HWF utilizado no presente trabalho, faz-se necessário transformar essas distâncias em distâncias longitudinais e laterais com relação ao agente móvel (robô) visto que as entradas do controlador *Fuzzy* I, destinado ao alvo, são as mesmas. Para isso, além da posição x e y do robô, é imprescindível conhecer o ângulo do robô em relação à origem do plano cartesiano.

Esta subseção apresenta o procedimento desenvolvido para esta transformação, considerando a título de representação, um caso no qual o robô se encontra na posição (0,0) formando 45 graus com o eixo “x”, e o alvo localizado na posição (200,115) conforme representado na Figura 6.

Observa-se que o alvo está localizado na frente e à direita do robô, independente do quadrante do plano cartesiano que o mesmo esteja. Caso o robô estivesse formando 0 grau com o eixo x do plano, o alvo estaria localizado à esquerda e na frente do robô. Torna-se por isso necessário quantificar essas distâncias.

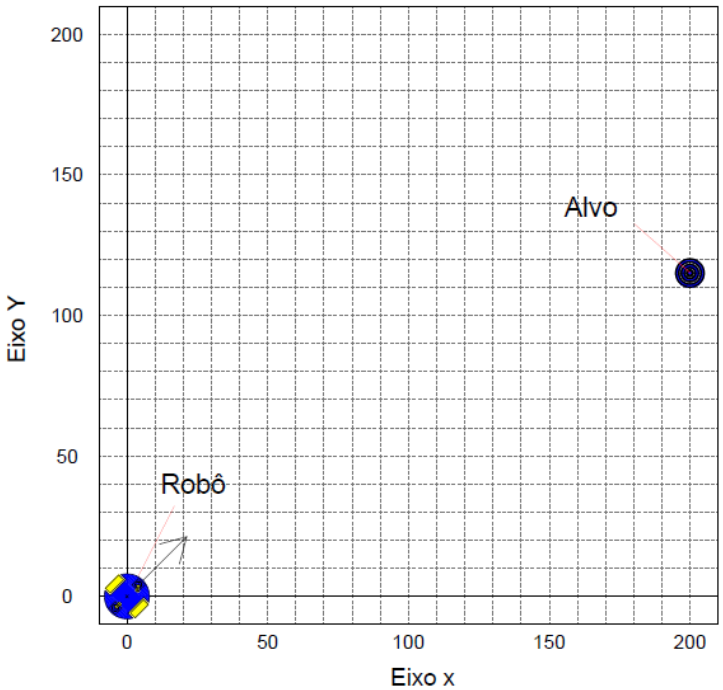
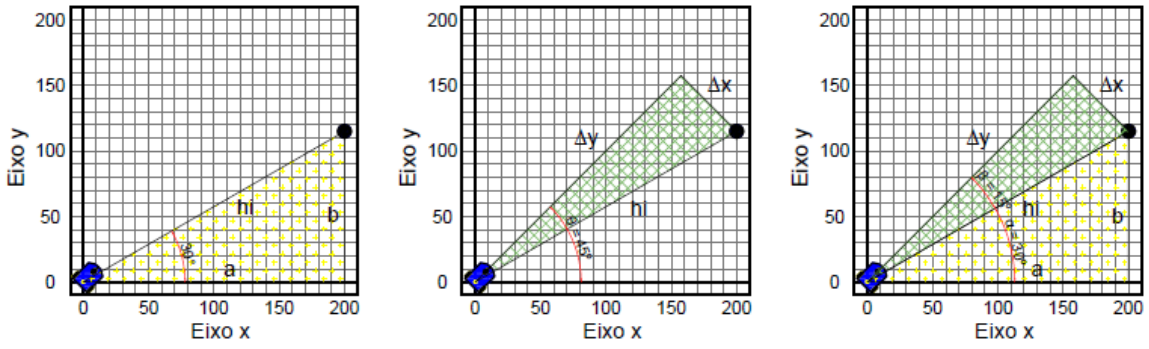


Figura 6 - Posicionamento alvo e robô

Fonte: Autoria própria.

Para esses cálculos será utilizado o teorema de Pitágoras nos triângulos apresentados nas figuras a seguir. A Figura 7a apresenta o triângulo retângulo formado entre o a posição inicial do robô (triângulo 1), o alvo e o eixo x. Na Figura 7b destaca-se o triângulo retângulo formado entre a posição inicial do robô (triângulo 2), o alvo e o eixo longitudinal do robô, e a Figura 7c representa a junção desses dois triângulos.



(a) (b) (c)

Figura 7 - Triângulos formados entre o robô e o alvo

Fonte: Autoria própria.

Nesta figura o ângulo θ representa o ângulo formado entre o eixo “x” e o robô, o ângulo α é formado entre o eixo x e a reta formada entre o ponto inicial e o ponto final (alvo), β é a diferença entre α e θ, “a” e “b” são catetos do triângulo 1, “hi” é a hipotenusa dos dois triângulos (sendo também a distância em linha reta entre o alvo e o robô) e “Δy” e “Δx”, os catetos do triângulo 2. É evidente que “Δy” e “Δx” também representam as distâncias longitudinal e lateral entre o robô e o alvo, respectivamente. Por fim, esses valores eram os quais se buscavam inicialmente.

Os valores “a” e “b” são conhecidos, visto que se localiza previamente o alvo e o robô, sendo assim os mesmos serão utilizados para o cálculo da hipotenusa:

$$a^2 + b^2 = hi^2 \tag{4}$$

$$hi = \sqrt{a^2 + b^2} \tag{5}$$

$$hi = \sqrt{200^2 + 115^2} \tag{6}$$

$$hi = 230{,}70 \quad (7)$$

Considerando o triângulo 2, pode-se afirmar que:

$$\operatorname{sen}\beta = \Delta x/hi \quad (8)$$

$$\cos\beta = \Delta y/hi \quad (9)$$

O que implica em:

$$\Delta x = sen\beta * hi \quad (10)$$

$$\Delta y = cos\beta * hi \quad (11)$$

Sendo assim, através de (5), (10) e (11) pode-se transformar distâncias uni-cartesianas em distâncias relativas independente do ângulo a qual o robô esteja.

Entretanto, só essas informações não bastam; é preciso também localizar o quadrante em que o robô se encontra.

A Figura 8 apresenta as regiões do plano cartesiano às quais o seno e cosseno são positivos.

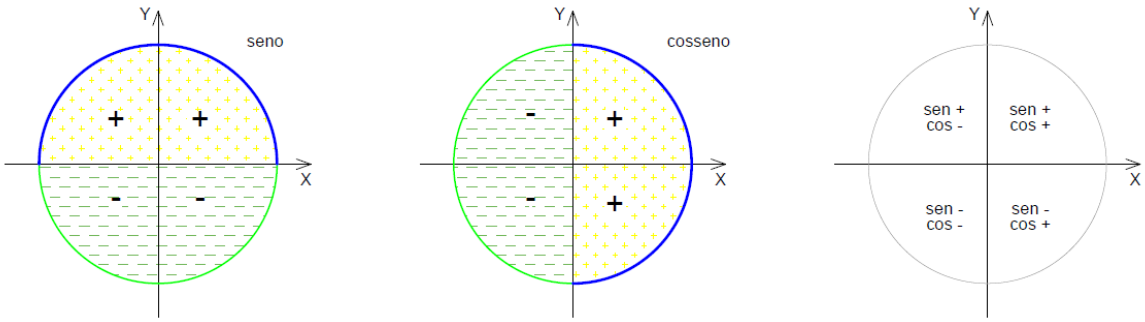


Figura 8 - Sinais do seno e cosseno

Fonte: Autoria própria.

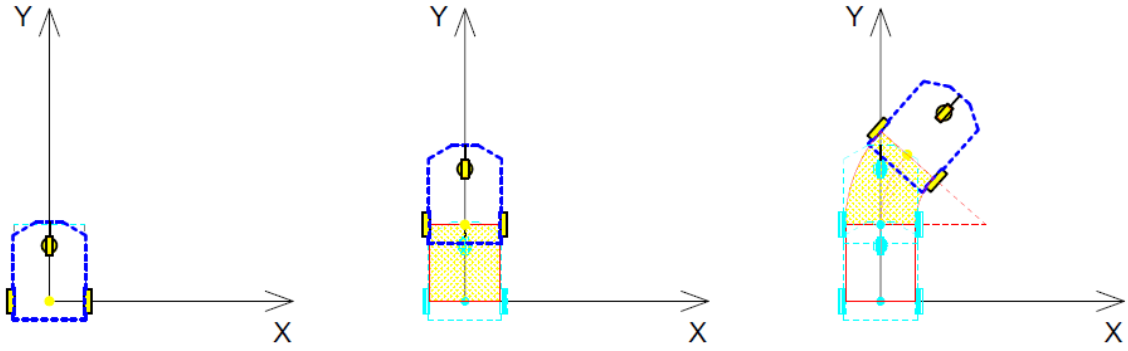
Sendo assim, de uma simples base de regras, pode-se descobrir em qual quadrante do plano cartesiano móvel do robô o agente se encontra. Se estiver no primeiro quadrante, implica que o alvo estará à direita e na frente do robô. Considerando o quadrante a qual esteja, será acrescentado um incremento ao α, para a obtenção de um ângulo correto. As regras são:

- Se cosseno de α é positivo, e o seno de α é positivo, não acrescenta incremento, o alvo está no primeiro quadrante;
- Se cosseno de α é negativo, e o seno de α é positivo, acrescenta-se um incremento de 180º; o alvo está no segundo quadrante;
- Se cosseno de α é negativo, e o seno de α é negativo, acrescenta-se um incremento de 180º; o alvo está no terceiro quadrante;
- Se cosseno de α é positivo, e o seno de α é negativo, acrescenta-se um incremento de 360º; o alvo está no quarto quadrante;

4.2.1.2 Transformando pulsos em movimento

As saídas do sistema HWF são pulsos nas rodas laterais. Então é preciso transformar essas respostas em movimentos no plano cartesiano para que o agente se localize a cada passo. Para isso, são apresentados nesta subseção os procedimentos para esses cálculos.

Considere a Figura 9 que representa o movimento do robô em dois passos. A Figura 9(a) representa a posição inicial do robô. No primeiro passo (b) foi considerada uma volta em cada roda, e no segundo passo (c), foram considerados dois pulsos na roda esquerda e um na roda direita.



(a) (b) (c)

Figura 9 - Trajetória do robô 1

Fonte: Autoria própria.

Para o primeiro passo, o procedimento para o deslocamento é simples, no qual será utilizada a fórmula do comprimento da circunferência para determinar o deslocamento, conhecendo previamente o raio das rodas do robô (no caso, 2,5 cm):

$$c = 2.\pi.R \quad (12)$$

$$c = 2.\pi.2{,}5 \quad (13)$$

$$c = 15{,}708\ cm \quad (14)$$

Sendo assim, o agente se deslocou 15,708 cm mantendo a mesma angulação em relação ao eixo x.

Para o passo 2, os pulsos nas rodas são diferentes, então se torna necessário outro procedimento para se chegar até o raio de curvatura e consequentemente, o deslocamento propriamente dito. A Figura 10 representa os movimentos realizados no passo 2 e suas representações angulares.

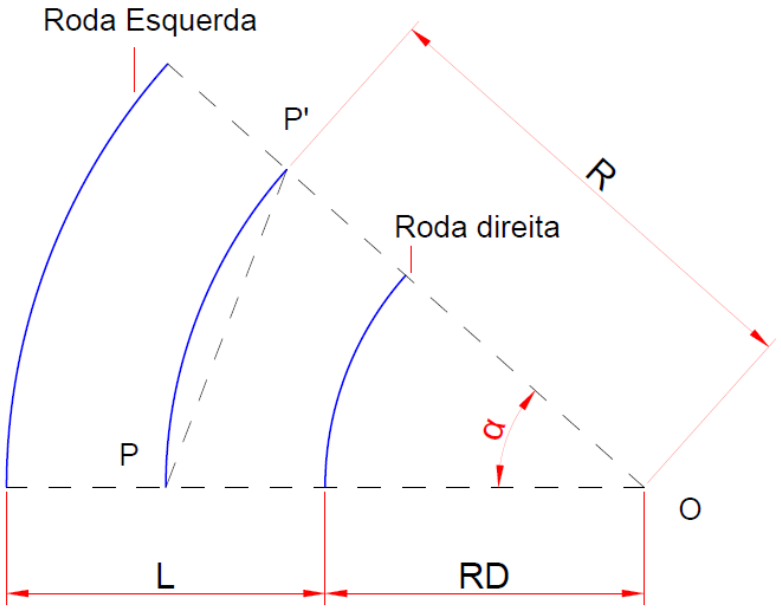


Figura 10 - Angulações

Fonte: Autoria própria.

Em que “P”, o ponto central do eixo do agente, “L” é a largura entre as rodas e “R” seria o raio de curvatura do robô, sendo considerado na roda direita. Ao calcular os deslocamentos, será considerado no programa o deslocamento do ponto “P”. O valor de “R” será considerado como a soma de “RD” com L/2, conforme apresentado na equação a seguir:

$$R = RD + L/2 \quad (15)$$

O raio de giro da roda esquerda corresponde à soma entre raio de giro da roda direita (R) e o da distância entre as rodas (L), ou seja:

$$RE = RD + L \quad (16)$$

Serão apresentados quatro casos envolvendo relações diferentes entre pulso na roda esquerda e roda direita para então chegar a uma fórmula geral que será utilizada no programa.

No primeiro caso, “RE” (raio de giro da roda esquerda) representa o dobro de “RD” (duas vezes o raio de giro da roda direita). Representando matematicamente:

$$RE = 2.RD \quad (17)$$

De (16) e (17), obtêm-se:

$$RD + L = 2.RD \quad (18)$$

$$RD - 2.RD = -L \quad (19)$$

$$RD = L \quad (20)$$

No segundo caso, “RE” (raio de giro da roda esquerda) representa o triplo de “RD” (três vezes o raio de giro da roda direita). Representando matematicamente:

$$RE = 3.RD \quad (21)$$

De (16) e (21), obtêm-se:

$$RD + L = 3.RD \quad (22)$$

$$RD - 3.RD = -L \tag{23}$$

$$RD = L/2 \tag{24}$$

No terceiro caso, “RE” (raio de giro da roda esquerda) representa o quádruplo de “RD” (quatro vezes o raio de giro da roda direita). Representando matematicamente:

$$RE = 4.RD \tag{25}$$

De (16) e (25), obtêm-se:

$$RD + L = 4.RD \tag{26}$$

$$RD - 4.RD = -L \tag{27}$$

$$RD = L/3 \tag{28}$$

No quarto caso, “RE” (raio de giro da roda esquerda) representa a metade de “RD” (duas vezes o menor raio de giro da roda direita). Representando matematicamente:

$$RE = (1/2).RD \tag{29}$$

De (16) e (29), obtêm-se:

$$RD + L = (1/2).RD \tag{30}$$

$$RD - (1/2).RD = -L \tag{31}$$

$$RD = L/(-1/2) \tag{32}$$

Logo, nota-se que o raio de giro do robô está diretamente relacionado à distância entre as duas rodas, e inversamente proporcional à porcentagem de giro “x” em que RE é maior que RD. Para calcular essa porcentagem de giro em que RE é maior que RD, foi utilizado o seguinte método: Primeiramente foi considerado RD como cem por cento. Se for acrescentada uma porcentagem “x” deste RD, obtém-se RE:

$$RE = RD + x.RD \tag{33}$$

Isolando “x”:

$$x = (RE - RD)/RD \tag{34}$$

Ou seja,

$$x = \left(\frac{RE}{RD}\right) - 1 \tag{35}$$

Como RE está relacionado ao pulso we fornecido pelo controlador e RD está relacionado ao pulso wd fornecido pelo controlador, pode-se afirmar que a relação (RE/RD) é igual a (we/wd) e serão utilizados esses termos para o cálculo do raio R, visto que são esses valores que temos inicialmente. Logo:

$$x = \left(\frac{we}{wd}\right) - 1 \tag{36}$$

Como R é inversamente proporcional à porcentagem de giro de uma roda em relação à outra (*x*), generalizando, pode-se afirmar que:

$$RD = \frac{L}{[\left(\frac{we}{wd}\right) - 1]} \tag{37}$$

Sendo “we” o valor do pulso na roda esquerda e “wd”, o valor do pulso na roda direita. Valores esses, fornecidos pelo controlador, ainda considerando o movimento da roda direita, pode-se observar que para o passo 2, a roda direita deu

uma volta, o que implica que se pode obter através da fórmula do comprimento do círculo, seu deslocamento e fazer o mesmo processo para a roda esquerda, considerando o dobro desse valor, visto que deu duas voltas. Para este procedimento, serão utilizados dois conceitos básicos: A fórmula do comprimento do círculo (38) e comprimento do arco de uma circunferência (39), visto que o comprimento do arco RD corresponde à distância percorrida pela roda direita, sendo considerada uma volta na mesma.

$$c_d = 2.\pi.r \tag{38}$$

$$c_d = \frac{\alpha.\pi.R}{180} \tag{39}$$

Observe que as voltas nas rodas equivalem ao comprimento do arco proporcionado pelas mesmas. As variáveis conhecidas são: “L”, “r”, “we” e “wd”. A partir de (38) pode-se obter o valor de “c_d”. Aplicando esse valor em (39), obtém-se o valor de “RD”.

Reorganizando os termos de (39) de modo a isolar α, tem-se:

$$\alpha = \frac{180.c_d}{\pi.R} \tag{40}$$

O valor de R é apresentado em (15).

Agora se tem os valores necessários para descobrir o deslocamento do ponto “P” bem como sua direção. O procedimento será feito a partir do triângulo formado por “P”, “P’” e “O”, representado nas Figura 11:

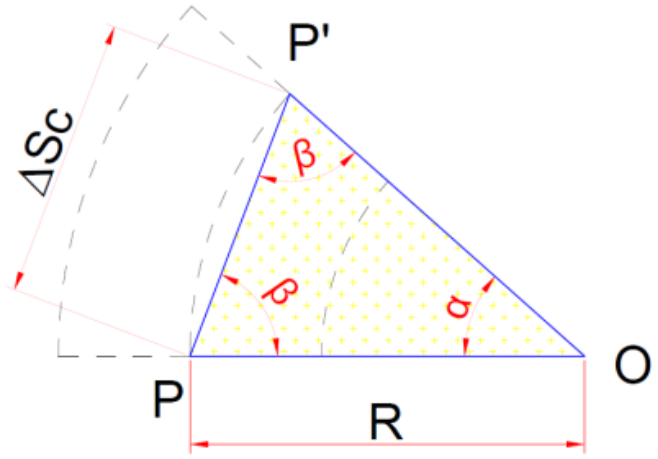


Figura 11 - Triângulo PP'O

Fonte: Autoria própria.

Tem-se um triângulo isósceles na qual os valores R e α são conhecidos. O cálculo dos valores de β possibilitará conhecer a direção em que o ponto P se desloca, e o cálculo de valor de ΔSc (deslocamento efetivo do robô) possibilitará conhecer o módulo desse deslocamento.

A geometria plana pressupõe que a soma dos ângulos internos de um triângulo vale 180º. Para o presente caso;

$$\alpha + \beta + \beta = 180 \tag{41}$$

$$2.\beta = 180 - \alpha \tag{42}$$

$$\beta = \frac{(180 - \alpha)}{2} \tag{43}$$

Baseado na lei dos senos, o valor de ΔSc pode ser calculado.

$$\frac{R}{sen\beta} = \frac{\Delta Sc}{sen\alpha} \tag{44}$$

Sendo assim;

$$\Delta Sc = \frac{R.sen\alpha}{sen\beta} \tag{45}$$

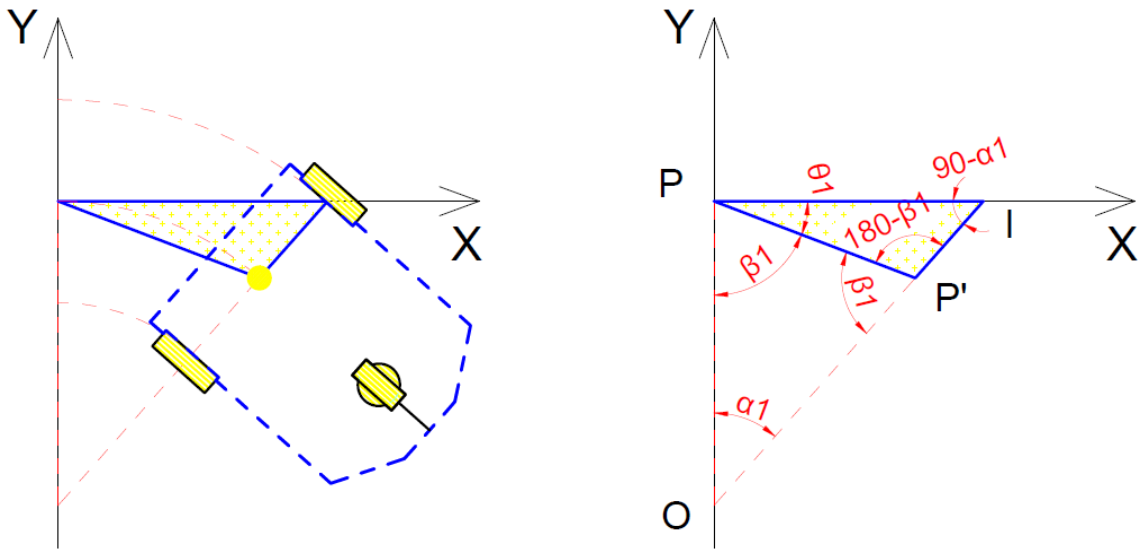
Decompondo esse deslocamento nas direções x e y do plano cartesiano e admitindo que θ∆s é o ângulo do sentido do deslocamento em relação ao eixo x.

$$\Delta Scx = \Delta Sc \,.\cos\theta \tag{46}$$

$$\Delta Scy = \Delta Sc \,.sen\theta \tag{47}$$

Entretanto é notável que o ângulo do deslocamento não seja igual ao ângulo ao qual o robô estará no fim do passo (caso os pulsos nas rodas sejam diferentes), visto que a trajetória é uma curva e os deslocamentos parciais que foram apresentados no cálculo anterior. Sendo assim, seguem os cálculos para a obtenção ângulo do robô em cada passo.

A Figura 12(a) representa o movimento do robô iniciando no ponto (0,0), ao qual foram enviados dois pulsos na roda esquerda e um pulso na roda direita. O triângulo destacado na Figura 12(b) é formado pela reta do deslocamento do robô (P-P'), a projeção da linha entre o ponto "O" (centro de giro do movimento) e o eixo "X":



(a) (b)

Figura 12 - Triângulo PP'I

Fonte: Autoria própria.

Ao fazer a soma dos ângulos internos desse triângulo, é obtida a seguinte equação:

$$180 = \theta 1 + (180 - \beta 1) + (90 - \alpha 1) \quad (48)$$

$$-\theta 1 = -180 + (180 - \beta 1) + (90 - \alpha 1) \quad (49)$$

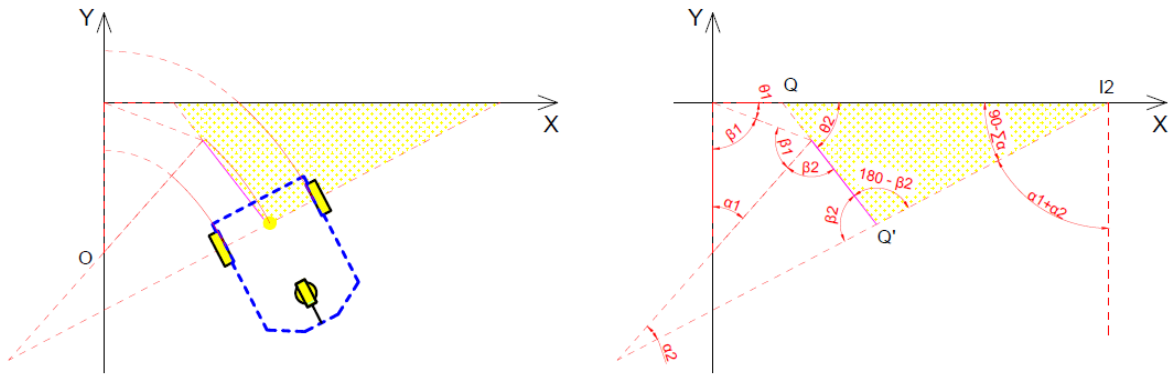
$$-\theta 1 = -180 + 180 - \beta 1 + 90 - \alpha 1 \quad (50)$$

$$-\theta 1 = 90 - \beta 1 - \alpha 1 \quad (51)$$

Como o giro de θ foi no sentido anti-horário, o valor do mesmo foi negativo.

Continuando a movimentação do robô, ao se fazer um segundo passo, pode-se obter o triângulo destacado na Figura 13.

.



(a) (b)

Figura 13 - Triângulo QQ'I2

Fonte: Autoria própria.

Ao fazer a soma dos ângulos internos desse triângulo, é obtida a seguinte equação:

$$180 = \theta 2 + (180 - \beta 2) + (90 - \Sigma\alpha) \quad (52)$$

$$-\theta 2 = -180 + (180 - \beta 2) + (90 - \Sigma\alpha) \quad (53)$$

$$-\theta 2 = -180 + 180 - \beta 2 + 90 - \Sigma\alpha \quad (54)$$

$$-\theta 2 = 90 - \beta 2 - \Sigma\alpha \quad (55)$$

Foram feitas várias simulações para demonstrar a consistência das fórmulas de obtenção dos ângulos parciais que a agente forma em relação ao eixo “x”. Todas as simulações feitas no trabalho (inclusive as não demonstradas) condisseram com os cálculos apresentados acima.

Os “α’s” representam a soma de todos os ângulos de giro do robô em relação ao ângulo inicial, de modo que, quando o robô gira para a esquerda este valor é negativo. Sendo assim, ao acrescentar à equação (55) o ângulo inicial e generalizar os termos, obtém-se assim a equação que informará a direção do robô em cada passo.

$$\theta i = 90 - \beta i - \Sigma\alpha + \theta 1 \quad (56)$$

4.3 MODELO DE MOVIMENTO

Baseado nos cálculos apresentados na subseção 4.2, o modelo de movimentação do robô implementado no simulador é apresentado a seguir:

O robô se encontra a uma determinada distância longitudinal e lateral do alvo:

1 - O simulador calcula qual é essa distância e determina o quadrante que esse alvo se encontra através das equações:

$$\Delta x = sen\beta * hi \quad (57)$$

$$\Delta y = cos\beta * hi \quad (58)$$

E das regras baseadas em senos e cossenos impostas para determinar o quadrante correto, utilizando a adição de um incremento ao ângulo. Baseado nessas

distâncias, o sistema HWF apresenta a resposta dos pulsos nos motores, sendo o *Fuzzy* I para o caso de os sensores não captarem alvos e *Fuzzy* II para o caso em que captarem.

2 - Para transformar os pulsos em movimento, caso os pulsos sejam iguais, foi utilizada a formula baseada na equação (12):

$$c = 2.\pi.R \tag{59}$$

Caso os pulsos sejam diferentes, obtêm-se o raio de giro através da fórmula:

$$R = \frac{L}{[\left(\frac{we}{wd}\right) - 1]} + \frac{L}{2} \tag{60}$$

Em termos de ângulos, se o pulso nas duas rodas for o mesmo, θ é igual ao θ anterior. Caso os pulsos sejam diferentes, recorre-se à fórmula:

$$\theta i = 90 - \beta i - \Sigma\alpha + \theta 1 \tag{61}$$

Em que $\alpha = 180.\mathrm{c_d}/\pi.\mathrm{R}$, $\mathrm{c_d} = \alpha.\pi.\mathrm{R}/180$ e $\beta = (180 - \alpha)/2$.

Obtendo assim o raio de giro, e a angulação final.

3 - Por fim, ao se calcular os deslocamentos utilizam-se:

$$\Delta Sc = \frac{R.sen\alpha}{sen\beta} \tag{62}$$

Para calcular os deslocamentos em x e y:

$$\Delta Scx = \Delta Sc\,.\cos\theta \tag{63}$$

$$\Delta Scy = \Delta Sc\,.sen\theta \tag{64}$$

Este simulador foi desenvolvido para o robô utilizado, deste modo, qualquer arquitetura implementada será simulada de forma coerente, baseando-se nos pulsos nas rodas. Os códigos do simulador implementado no MatLab envolvendo as duas arquiteturas utilizadas são apresentados nos APÊNDICES A e B, respectivamente.

5 SISTEMAS DE NAVEGAÇÃO AUTÔNOMOS DESENVOLVIDOS

5.1 ARQUITETURA SAPHIRA

Dentre as arquiteturas híbridas conhecidas, este trabalho se baseia na arquitetura conhecida como Saphira, apresentada em Konolige e Myers (1998). Uma arquitetura baseada em comportamentos, na qual o diferencial em relação às demais arquiteturas é o fato dos comportamentos serem escritos e combinados utilizando técnicas baseadas em lógica nebulosa (SAFFIOTTI, 1997).

A Figura 14 apresenta o diagrama esquemático da arquitetura Saphira.

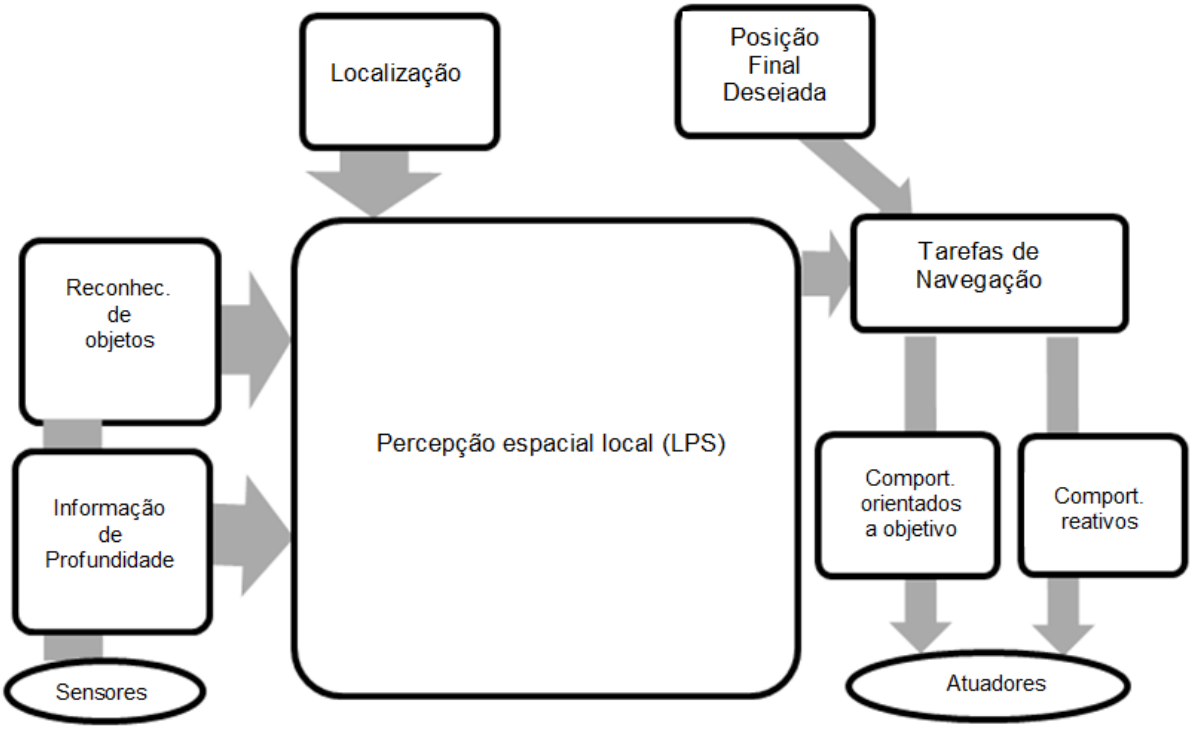


Figura 14 - Arquitetura clássica *Saphira*
Fonte: Romero (2014).

Nesta figura, o bloco reconhecimento de objetos refere-se aos sinais nos sensores e informação de profundidade é a quantificação destes sinais, convertendo-os em medidas em centímetros. A arquitetura tem como componente central uma representação do ambiente ao redor do robô chamada de LPS (*Local Perceptual Space),* que auxilia tanto na reatividade quanto nas ações deliberativas

do sistema. A maior parte das ações é planejada e executada tendo como base esse conhecimento. No âmbito das tarefas de navegação são analisados os dados de entrada e faz-se então a seleção entre comportamentos orientados ao objetivo e comportamentos reativos.

A arquitetura Saphira assume que o robô é uma plataforma móvel capaz de oferecer serviços básicos utilizando um protocolo específico.

5.2 ARQUITETURA HIERÁRQUICA

Quando o Controlador *Fuzzy* tem que lidar com problemas de maior complexidade como é o caso de problemas multi objetivos, uma solução amplamente aceita na literatura é decompor o problema do controlador de forma hierárquica, utilizando modelos computacionais de baixa dimensão. Uma arquitetura hierárquica baseada em decomposição foi originalmente proposta por Brooks (1986) e se utilizava de máquinas de estado para modelar os comportamentos do robô em diversos níveis.

A arquitetura proposta neste trabalho, denominada HWF (*Hierarchical Fuzzy Weighted*) assemelha-se a Saphira, para a modelagem de suas ações hierárquicas e reativas. Neste caso, o que corresponderia à base de regras LPS é substituída por uma base de regras *Fuzzy* hierarquicamente organizada de acordo com as tarefas e prioridades a serem cumpridas. Já o comportamento reativo é inspirado na arquitetura de subsunção apresentado por Brooks (1986).

A arquitetura de Subsunção trabalha tomadas de decisões hierárquicas em paralelo: recebe estímulos sensoriais ao mesmo tempo e deve estar apta a gerenciar os objetivos de acordo com prioridades e níveis hierárquicos. Isso se deve a necessidade de dois ou mais objetivos principais e contraditórios, como por exemplo, atingir um alvo e desviar de obstáculos presentes nas trajetórias precisarem ser atingidos ao mesmo tempo. Desse modo, se faz necessário ter uma hierarquia de ações em problemas de navegação robótica.

5.3 ARQUITETURA *FUZZY* HIERÁRQUICA - HWF

A arquitetura *Fuzzy* proposta tem duas bases de regras, as quais trabalham de maneira independente e com níveis de competências diferenciados e hierárquicos. Uma característica herdada da arquitetura clássica de subsunção está na seguinte definição: quando um nível determinado de competência (ou objetivo) está habilitado, inibe a ação de outro. Desse modo, o Controlador *Fuzzy* executa tomadas de decisões com nível de competência, priorizando o maior nível (desvio de obstáculo é mais importante do que coleta de alvo) (GOERICK, 2011). Com isso, quando o agente autônomo realiza o objetivo de maior prioridade (desviar de obstáculos), os outros objetivos são temporariamente desativados (o robô deixa de perseguir o alvo) e assim, um mecanismo de chaveamento deve ser implementado. A Figura 15 mostra Arquitetura *Fuzzy* hierárquica desenvolvida.

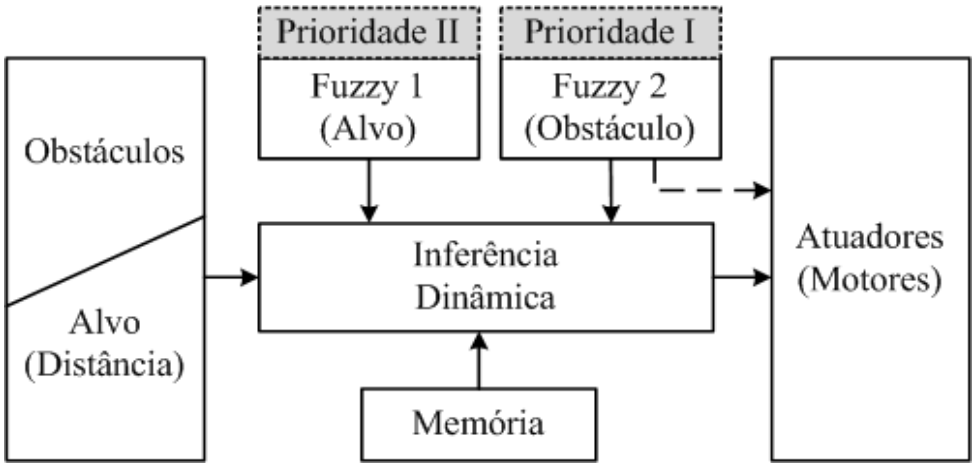


Figura 15 – Arquitetura *Fuzzy* Hierárquica

Fonte: Adaptado de Mendonça (2011).

Cada um dos blocos da Figura 15 será detalhado a seguir: O primeiro bloco (Obstáculos / Alvo) representa as variáveis de entrada, se algum sensor captou obstáculos, essas variáveis serão os sinais nos sensores. Caso não haja obstáculos, as variáveis serão a distância lateral e frontal entre o robô e o alvo. O bloco inferência dinâmica tem a função de definir de forma dinâmica qual será o controlador utilizado, baseado no tipo de entrada que este recebeu (sensor ou distâncias). O bloco memória refere-se à memória de localização do robô, para o

cálculo de sua posição atual em relação ao alvo. O bloco *Fuzzy* I representa o controlador direcionado ao alvo, de prioridade II, e o bloco *Fuzzy* II representa o controlador direcionado ao desvio de obstáculos. Por fim, o bloco dos atuadores representa a resposta dos controladores indicando os pulsos que devem ser aplicados pelos atuadores nas rodas laterais.

5.3.1 Controlador Para Perseguição de Alvos

O controlador lógico *Fuzzy* I tem como objetivo determinar a rota mais curta entre o ponto atual do robô e o alvo. A sua base de regras foi implementada a partir da observação heurística do comportamento dinâmico do agente em cenários virtuais.

A estratégia implementada consiste em abstrair os dados de entrada (adquiridos pelos sensores) e convertê-los em variáveis linguísticas (base de regras da programação) para então gerar uma resposta (pulsos nos motores acoplados às rodas esquerda e direita).

A Figura 16 contém o diagrama esquemático deste controlador. Nesta figura estão indicadas as variáveis de entrada relativas às distâncias entre o robô e o alvo, distância lateral (DSx) e distância frontal (DSy), admitindo como referência os sistemas de coordenadas do protótipo. E as variáveis de saídas, we e wd, que estão relacionadas aos pulsos enviados aos atuadores do robô.

Em especial, o universo de discurso das variáveis de entrada varia entre -1 e 1. Quando a distância lateral é negativa o alvo está localizado à esquerda do robô e quando positiva o alvo é localizado à direita. De forma análoga, quando a distância frontal é negativa, o alvo está localizado na parte de trás do robô e quando positiva, o alvo está localizado na parte da frente. Os sinais we e wd são pulsos nos motores acoplados às rodas esquerda e direita, respectivamente.

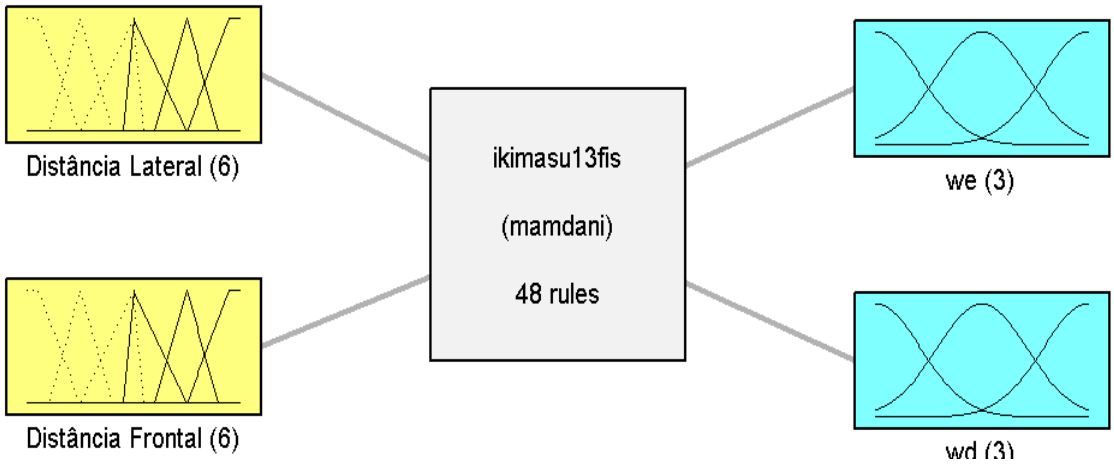


Figura 16 - *Fuzzy* I - Alvo

Fonte: Autoria própria.

O retângulo central na Figura 16 corresponde à base de regras, que representa o conhecimento heurístico sobre a movimentação do robô em busca do alvo, fazendo a combinação entre todas as funções de pertinência. A combinação das doze funções de pertinência (seis para cada entrada) totaliza 36 regras, foram considerados também os casos em que o alvo se localizava exatamente na frente do robô, atrás do robô, à esquerda do robô e à direita do robô. Considerando então mais três funções de pertinência (longe / médio / perto) para cada uma das quatro direções, o que compõe 12 regras a mais. Deste modo o número total de regras para este controlador *Fuzzy* foi 48 regras.

As Figuras 17 e 18 apresentam as funções de pertinência para as variáveis do controlador *Fuzzy* I envolvendo os dados de entrada:

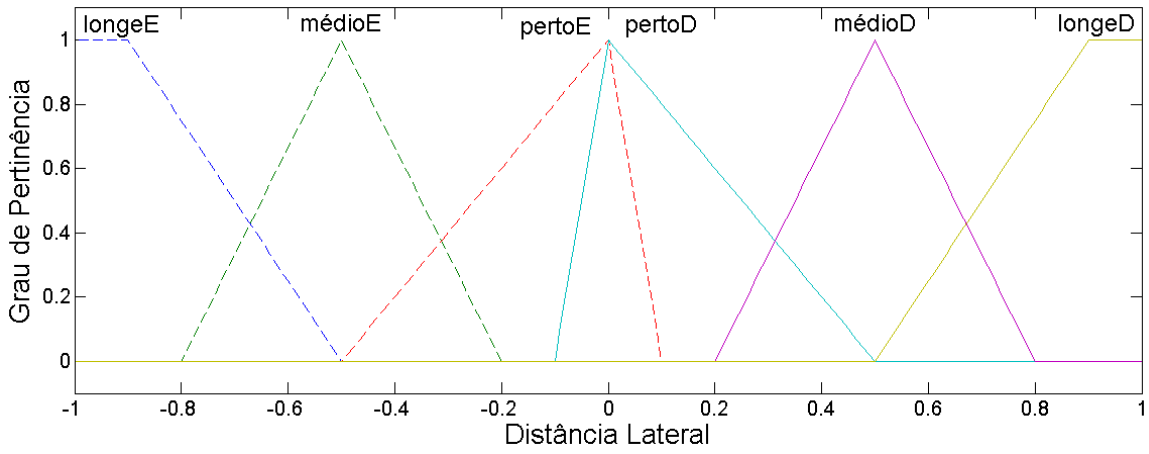


Figura 17 - Funções de pertinência de entradas 1 – *Fuzzy* I

Fonte: Autoria própria.

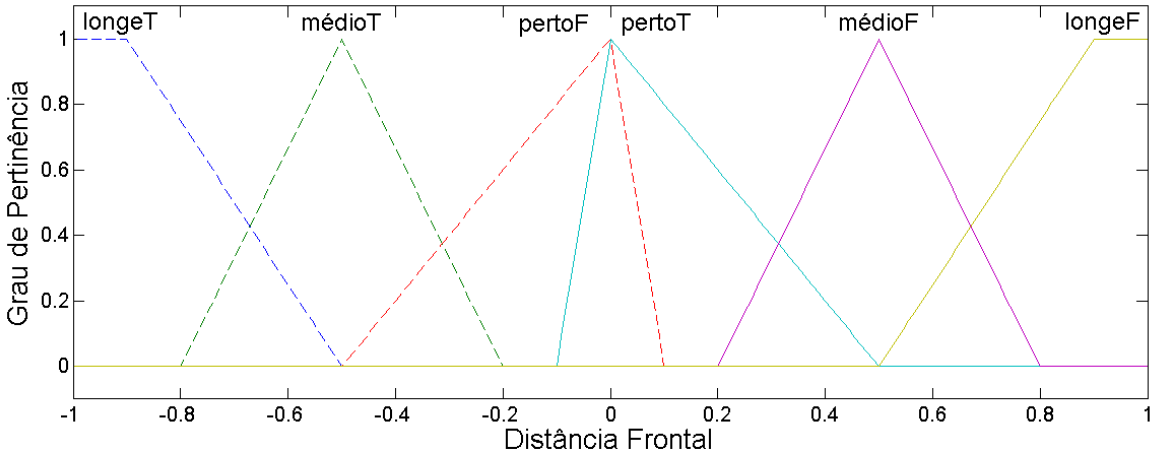


Figura 18 - Funções de pertinência de entradas 2 – *Fuzzy* I

Fonte: Autoria própria.

O universo de discurso de todas as funções de pertinência de saída está compreendido entre -1 e 1 visto que é utilizado um fator de escala para todas as entradas que divide o valor das entradas pelo valor máximo atingido.

As Figuras 19 e 20 apresentam as funções de pertinência para o *Fuzzy* I envolvendo os dados de saída:

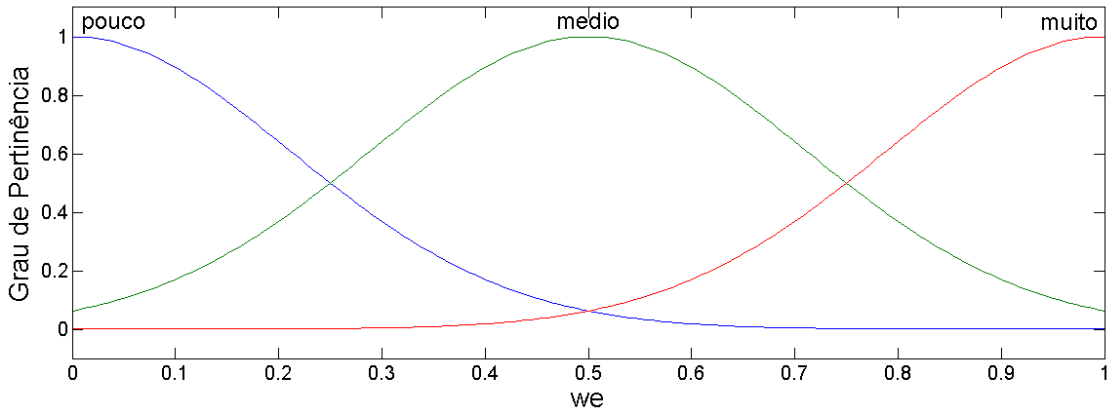
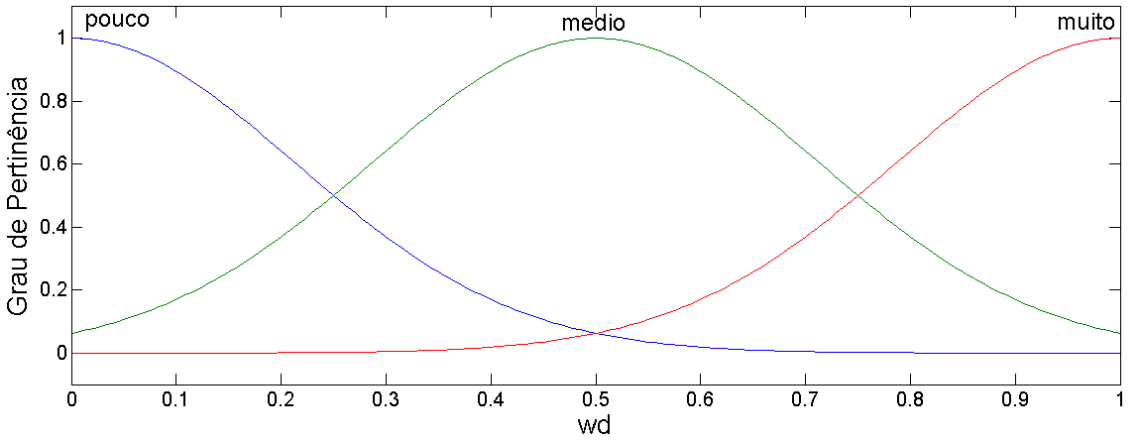


Figura 19 - Funções de pertinência de saídas I – *Fuzzy* I

Fonte: Autoria própria.



(b)

Figura 20 - Funções de pertinência de saídas 2 – *Fuzzy* I

Fonte: Autoria própria.

Seis exemplos das regras do controlador *Fuzzy I* são apresentados na Tabela 1.No apêndice A, consta a tabela completa.

Tabela 1. Exemplo de regras do *Fuzzy* I

1. Se (Distância Lateral é longe a esquerda) e (Distância Frontal é longe a frente) então (we é médio) (wd é alto) (0.5)
2. Se (Distância Lateral é longe a esquerda) e (Distância Frontal é média a frente) então (we é baixo) (wd é alto) (0.5)
3. Se (Distância Lateral é longe a esquerda) e (Distância Frontal é pouca a frente) então (wd é alto) (0.5)
4. Se (Distância Lateral é médio a esquerda) e (Distância Frontal é longe a frente) então (we é médio) (wd é alto) (0.5)
5. Se (Distância Lateral é médio a esquerda) e (Distância Frontal é média a frente) então (we é baixo) (wd é alto) (0.5)
6. Se (Distância Lateral é médio a esquerda) e (Distância Frontal é pouca a frente) então (wd é alto) (0.5)

Nas Figuras 21 e 22 são apresentadas as superfícies de controle do *Fuzzy* I correspondente às rodas esquerda (Figura 19) e direita (Figura 20).

O eixo x representa a distância lateral entre o robô e o alvo (DSx), o eixo y, a distância frontal entre os mesmos elementos (DSy) e o eixo z, os pulsos nas rodas: esquerda para a Figura 21 e direita para a Figura 22.

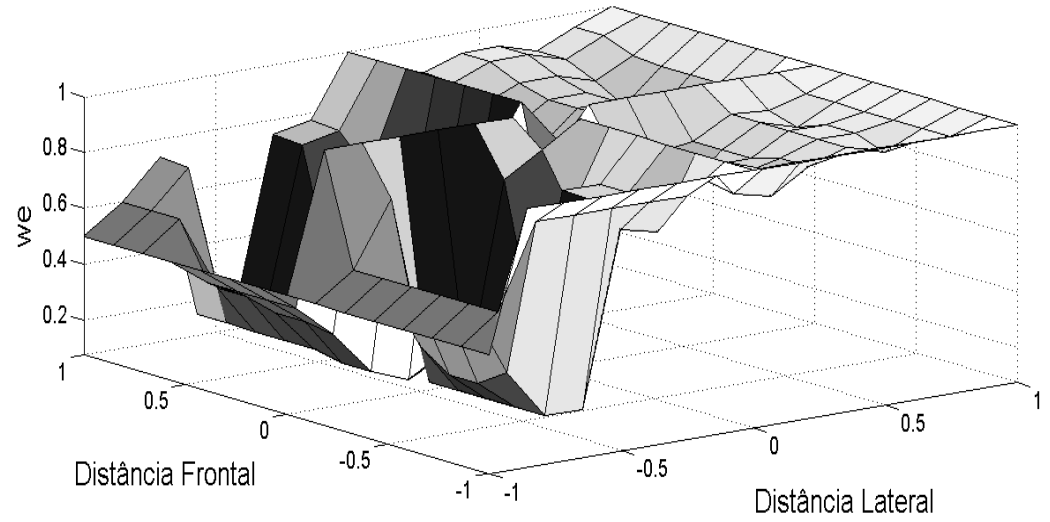


Figura 21 - Superfície de controle *Fuzzy* I (a)

Fonte: Autoria própria.

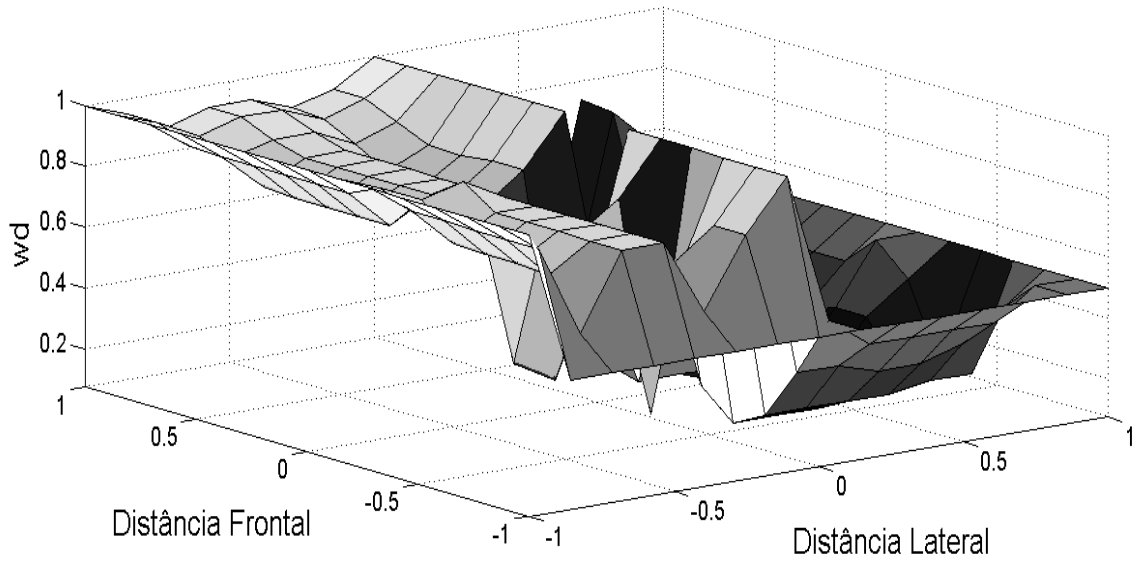


Figura 22 - Superfície de controle *Fuzzy* I (b)

Fonte: Autoria própria.

5.3.2 Controlador Para o Desvio de Obstáculos

Quando um obstáculo é detectado pelos sensores o sistema *Fuzzy* hierárquico priorizará o desvio de obstáculos através do controlador *Fuzzy* II. A percepção de algum obstáculo pelos sensores tem influência direta nos pulsos

enviados às rodas do agente. Caso não existam obstáculos, ele seguirá em direção ao alvo (ativando a base de regras do controlador *Fuzzy* I).

Em Rasshofer (Rasshofer e Gresser, 2005) compara-se o uso de sensores para medição de distância por meio de laser, ultrassom e radares de curto alcance. Baseado nessas comparações foi concluído que o mais viável, em termos de custo-benefício, seria o sensor de ultrassom para a construção do protótipo.

No protótipo, esses sensores serão calibrados para que enviem sinais a uma distância máxima de 15 cm, para que o agente não faça desvios desnecessários em sua rota (por localizar obstáculos além da área considerada).

A Figura 23 apresenta um diagrama esquemático do controlador *Fuzzy* II destinado apenas ao desvio de obstáculos. As entradas são as respostas dos sensores (esquerdo, direito e frontal, respectivamente) e as respostas "*we"* e "*wd"*, são os pulsos nas rodas esquerda e direita.

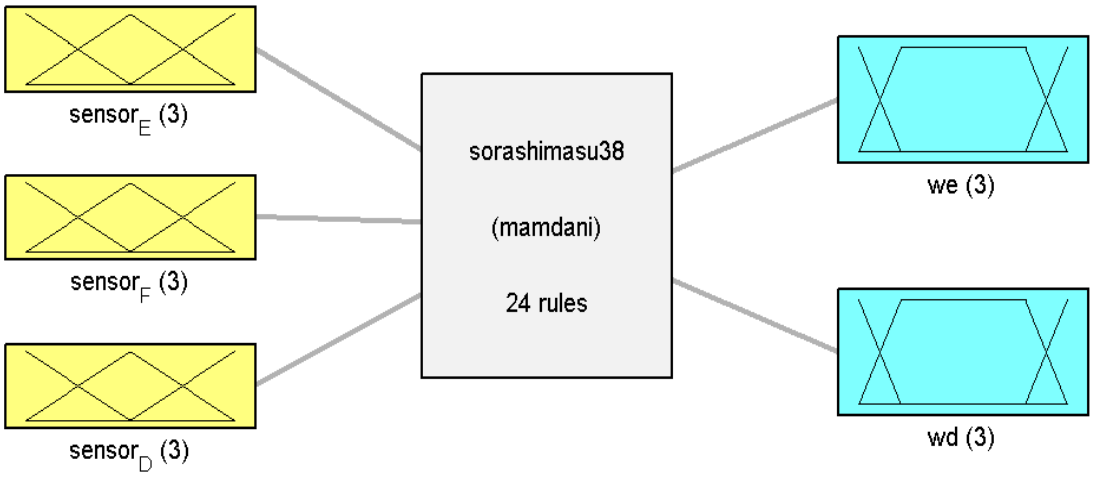


Figura 23 - *Fuzzy* II – Obstáculos
Fonte: Autoria própria.

O retângulo central na Figura 23 corresponde à base de regras, que representa o conhecimento heurístico sobre a movimentação do robô em busca do desvio de obstáculos, fazendo a combinação entre todas as funções de pertinência. A combinação das doze funções de pertinência (divididas em três sensores, sendo três para cada sensor) totaliza 27 regras, entretanto se de alguma forma os dois sensores laterais capitam obstáculos simultaneamente, isso significa "emboscada", deste modo o robô para e dá um giro de 90 graus para a direita. Foram inibidas

então as regras referentes à combinação dos sensores laterais. O controlador trabalha então com as seguintes combinações de sensores:

- Esquerdo e frontal – 9 regras;
- Direito e frontal – 9 regras;
- Apenas sensor esquerdo – 3 regras;
- Apenas sensor direito – 3 regras.

Totalizando deste modo 24 regras. Se de alguma forma apenas o sensor frontal captar obstáculos o robô fará um leve giro para a direita. Esta regra foi aplicada fora do ambiente do controlador *Fuzzy* para que esta regra não cause uma tendência de giro no robô para a direita.

As Figuras 24, 25 e 26 apresentam as funções de pertinência para o controlador *Fuzzy* II envolvendo os dados de entrada:

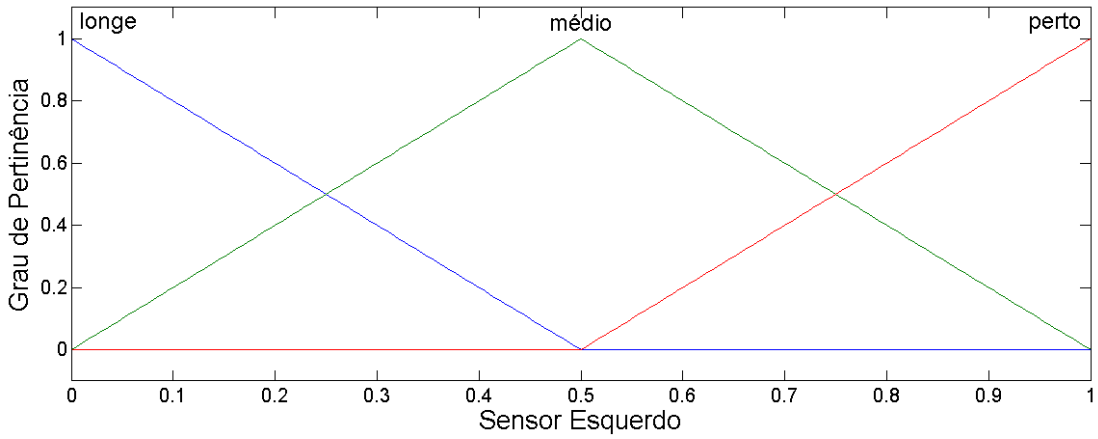


Figura 24 - Funções de pertinência de entradas 1 – *Fuzzy* II (a)

Fonte: Autoria própria.

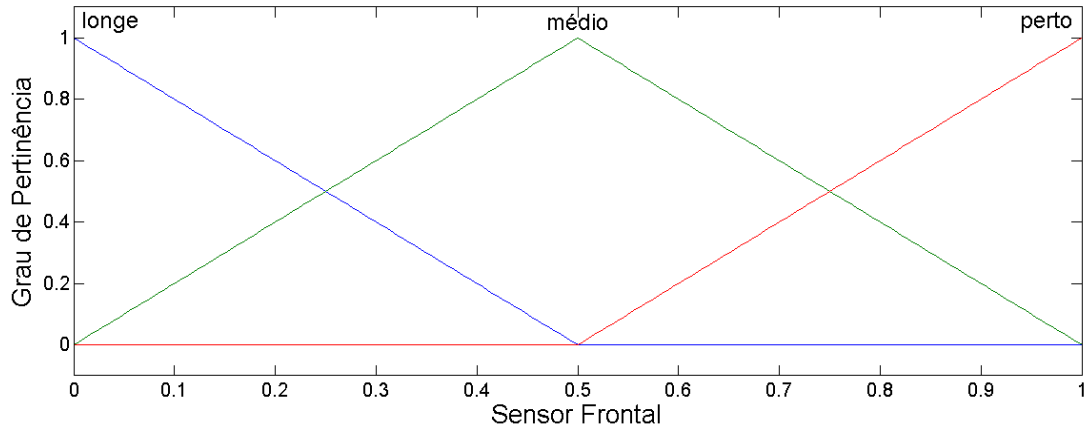


Figura 25 - Funções de pertinência de entradas 2 – Fuzzy II (b)

Fonte: Autoria própria.

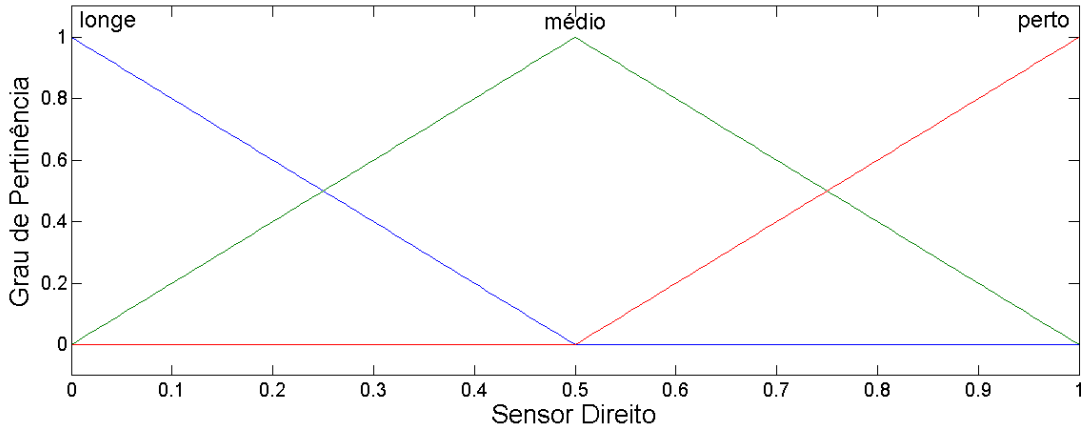


Figura 26 - Funções de pertinência de entradas 3 – Fuzzy II (c)

Fonte: Autoria própria.

As Figuras 27 e 28 apresentam as funções de pertinência para o *Fuzzy* II envolvendo os dados de saída:

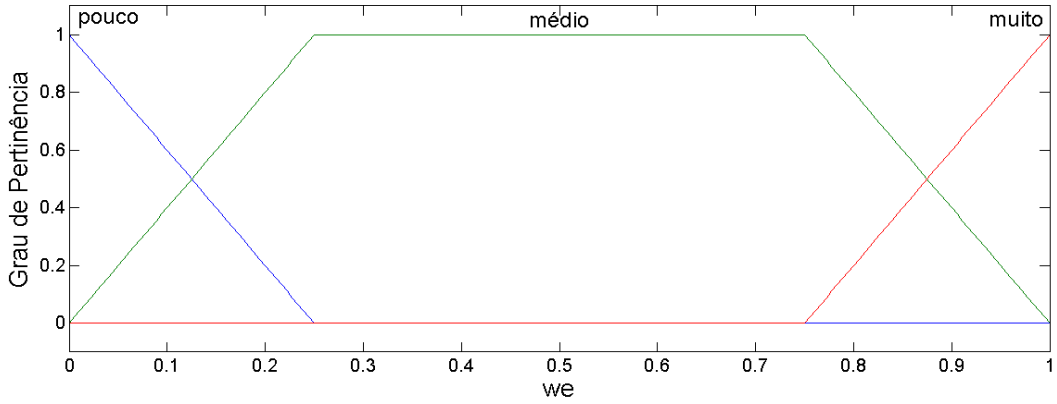


Figura 27 - Funções de pertinência de saídas 1 – *Fuzzy* II (a)

Fonte: Autoria própria.

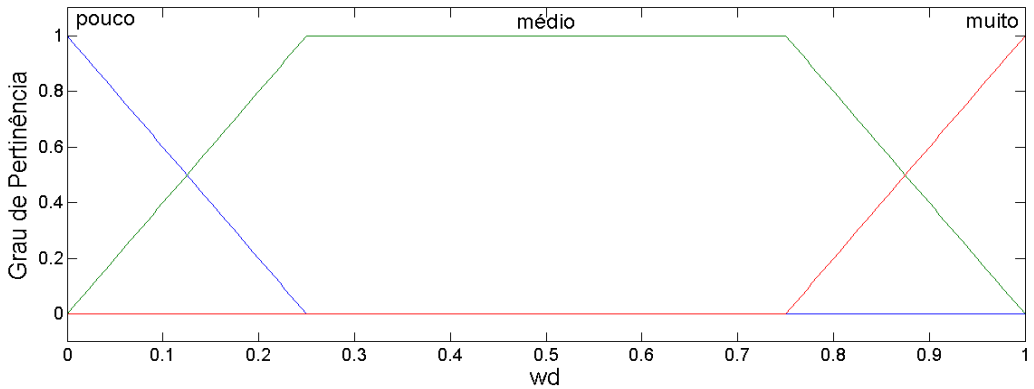


Figura 28 - Funções de pertinência de saídas 2 – *Fuzzy* II (b)

Fonte: Autoria própria.

Segue a Tabela 2 com seis exemplos de regras utilizadas no *Fuzzy* II para o desvio de obstáculos. A tabela completa encontra-se no Apêndice B.

Tabela 2. Exemplo de regras do *Fuzzy* II

1. Se (ind. sensor esquerdo é longe) e (ind. sensor frontal é longe) então (we é alto) e (wd é médio) (1)
2. Se (ind. sensor esquerdo é longe) e (ind. sensor frontal é médio) então (we é médio) e (wd é baixo) (1)
3. Se (ind. sensor esquerdo é longe) e (ind. sensor frontal é perto) então (we é alto) e (wd é baixo) (1)
4. Se (ind. sensor direito é longe) e (ind. sensor frontal é longe) então (we é médio) e (wd é alto) (1)
5. Se (ind. sensor direito é longe) e (ind. sensor frontal é médio) então (we é baixo) e (wd é médio) (1)
6. Se (ind. sensor direito é longe) e (ind. sensor frontal é perto) então (we é baixo) e (wd é alto) (1)

As superfícies de controle relacionadas ao controlador *Fuzzy* II são apresentadas nas Figuras 29 e 30. A Figura 29 refere-se à influência que a combinação dos sensores esquerdo e frontal exerce sobre a roda esquerda.

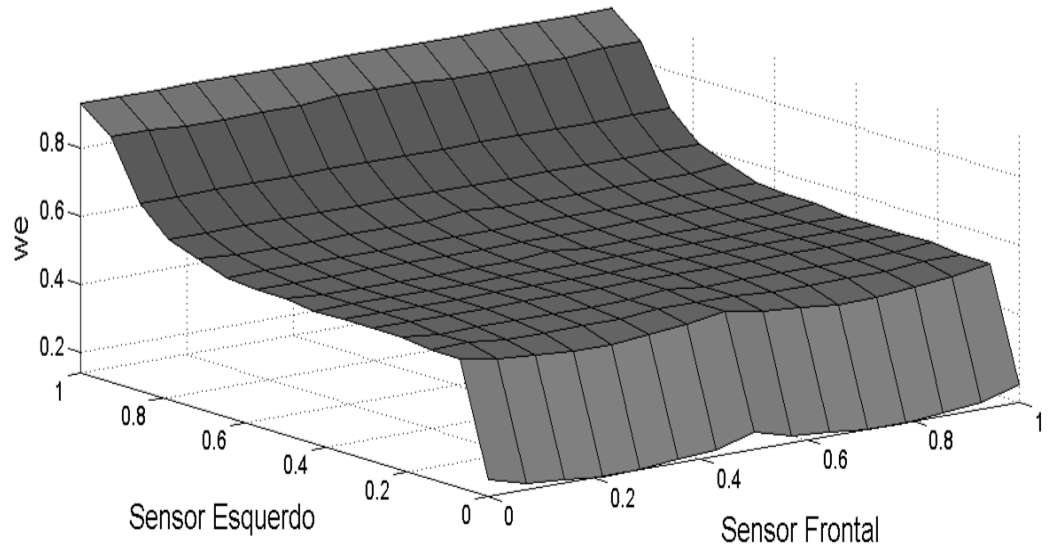
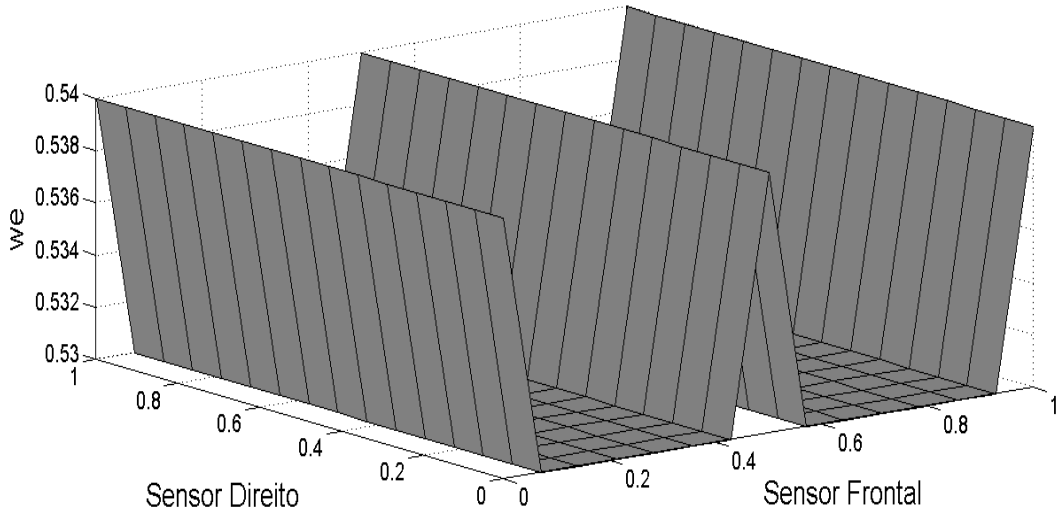


Figura 29 - Superfície de Controle *Fuzzy* II (a)

Fonte: Autoria própria.

A Figura 30 refere-se à combinação entre roda esquerda e os sensores direito e frontal como entrada. As regras são simétricas, ou seja, obtêm-se os mesmos resultados para a roda direita.



(b)

Figura 30 - Superfície de Controle *Fuzzy* II (b)
Fonte: Autoria própria.

Na Figura 30, o universo de discurso de “we” representando os pulsos na roda esquerda é [0.14 0.19] visto que quando há obstáculos na parte frontal direita do robô, o mesmo precisa virar para a esquerda, necessitando assim de um pulso baixo na roda esquerda.

Na Seção 6.1, esta arquitetura será aplicada na navegação autônoma de um robô em ambiente simulado.

5.4 ARQUITETURA HD-FCM

Arquiteturas híbridas em robótica procuram combinar características de abordagens reativas e deliberativas, buscando ao mesmo tempo reduzir as restrições no escopo de cada uma dessas características. Isto é, arquiteturas híbridas determinam o plano de ações do robô a partir de uma representação interna

global do conhecimento do mundo. Desta forma os objetivos do robô podem ser alcançados de forma eficiente (MENDONÇA *et al*., 2014).

Uma vez que as ações foram definidas, ao utilizar uma arquitetura híbrida no plano de ação, o robô pode responder rapidamente às mudanças dinâmicas no ambiente.

A arquitetura híbrida visa ser adequável para a resolução de problemas complexos atingindo metas de forma otimizada e eficiente em ambientes dinâmicos que requerem resposta rápida (LYONS; HENDRIKS,1995), (POLICASTRO; ROMERO, 2007) e (CALVO, 2007). Este tipo de arquitetura combina as principais características de abordagens reativas e deliberativas, conforme mostrado na Figura 31.

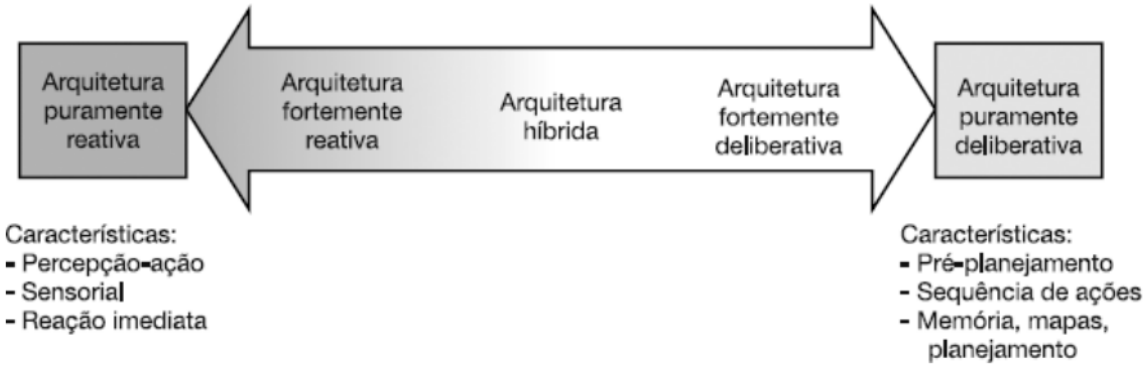


Figura 31 - Paradigma Híbrido

Fonte: Romero (2014).

As arquiteturas híbridas são desenvolvidas inspiradas no comportamento Planejador – Reator (LYONS, 1995). Neste caso, o reator consiste nos comportamentos reativos, ou seja, composições de sensores de percepção e respostas nos motores. O planejador é visto como um nível maior de sistema que gera um plano de ações a partir de níveis inferiores.

De modo específico, neste trabalho as duas arquiteturas desenvolvidas são fortemente reativas, comparadas, por exemplo, com a arquitetura de Subsunção de Brooks (1986). Entretanto, a arquitetura HD-FCM ainda agrega capacidade de adaptação que não é parte de arquiteturas reativas (MATARIĆ, 2007). De outro modo, a caracterização de uma arquitetura ser hibrida ou somente reativa não é uma solução fechada entre pesquisadores. Desse modo, para balizamento nessa dissertação a arquitetura HD-FCM é hierárquica, reativa ao sensoriamento, utiliza

mecanismo de troca de objetivos de modo semelhante ao HWF, entretanto, por possuir capacidade de adaptação a HD-FCM foi convencionada como uma arquitetura Híbrida.

A Figura 32 mostra a arquitetura HD-FCM (*Hybrid Dynamic - Fuzzy Cognitive Maps*). O sistema de percepção é a assimilação do ambiente por parte do robô, definindo qual o estado de operação, ou comportamento. Os estados modelados neste estudo são somente dois: Alcançar um alvo locado a uma posição previamente conhecida e desviar de obstáculos sem conhecimento prévio de suas localizações, através da percepção dos sensores e se encontram no bloco sistemas de estado interno.

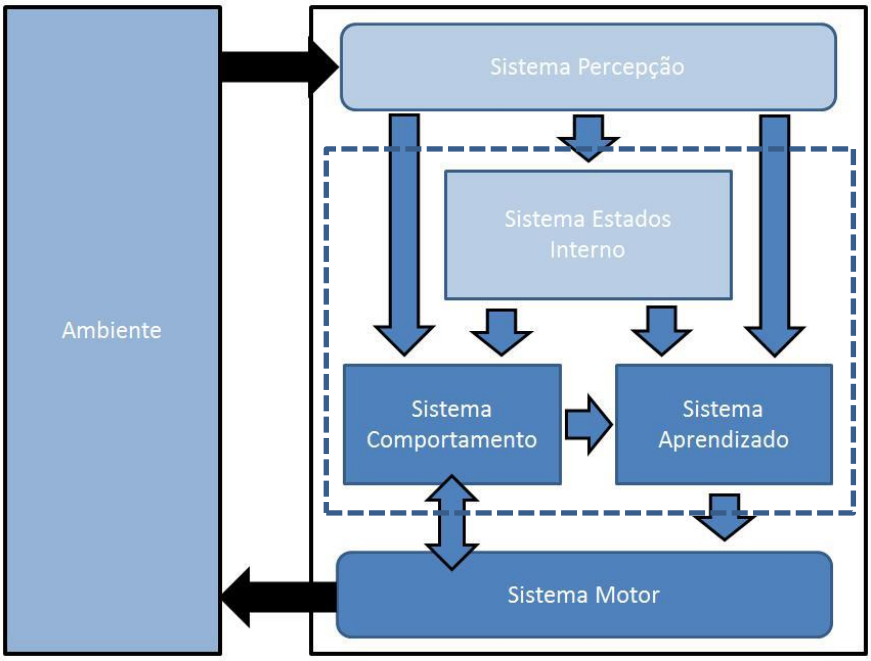


Figura 32 - Arquitetura híbrida HD-FCM
Fonte: Adaptado de Mendonça *et al* (2015)

A linha tracejada destaca o sistema Híbrido, e o sistema motor é responsável pelas interfaces de movimento do agente no ambiente de acordo com o estado atual, ou objetivo (desviar de obstáculos ou chegar ao alvo) dinamicamente ajustado pelo sistema de aprendizado. Este ajuste muda de acordo com os dados de entrada dos sensores (sistema de percepção) visando uma melhora nos resultados, e para isso utiliza-se um algoritmo de aprendizado de HEBB (1988).

A diferença básica entre o HD-FCM e o FCM clássico está nesta capacidade de ajuste dinâmico das relações casuais e alterações das variáveis FCM's que

correspondem ao estado interno corrente e que são chaveados por uma máquina de estados finita, de forma simples, usando lógica clássica para essa mudança de estados.

5.4.1 Controlador HD-FCM

O desenvolvimento do modelo HD-FCM (*Hybrid Dynamic - Fuzzy Cognitive Maps*) é inspirado no trabalho (MENDONÇA *et al*., 2013). Neste caso, foram utilizadas três entradas relacionadas à descrição do ambiente (presença de obstáculos) e duas saídas relacionadas aos movimentos do agente: pulsos nas rodas direita e esquerda.

Três decisões são originalmente modeladas, se o sensor esquerdo acusa obstáculo nesta posição, o veículo deve virar a direita e igualmente se o sensor direito acusa um obstáculo, o veículo vai para outro lado. A decisão de mudança de sentido implica em uma desaceleração leve. A terceira decisão é devida a um ambiente livre de obstáculos. Neste caso o robô segue em direção ao alvo.

A Figura 33 apresenta o robô (agente) para o modelo desenvolvido do HD-FCM, apresentando as distâncias ΔX, adotada como a distância lateral entre o agente e o alvo e ΔY, considerada a distância frontal entre o robô e o alvo. Nesta figura é mostrado o robô e o alvo inserido em um ambiente qualquer (cenário). Neste trabalho a posição do alvo é conhecida e o agente irá alternar entre dois FCM's (Figura 34 e Figura 35) para realizar respectivamente duas funções: Busca de alvos e desvio de obstáculos.

O HD-FCM I, apresentado na Figura 34, objetiva alcançar um ou mais alvos usando a distância previamente conhecida, em que "DSx" é a distância lateral entre o robô e o alvo (correspondendo a Δx na Figura 33) e "DSy" é a distância frontal (correspondendo a Δy na Figura 33). Se o alvo está localizado a esquerda do robô, "DSx" é negativo e se estiver localizado na parte de trás do robô, "DSy" é negativo. "LW" corresponde ao pulso na roda esquerda e "RW" corresponde ao pulso na roda direita. Os pesos W1-3 e W1-4 correspondem à influência que a distância lateral (DSX) tem nas rodas esquerda e direita, respectivamente, e os pesos W2-3 e W2-4

correspondem à influência que a distância frontal (DSY) possui nas rodas esquerda e direita, respectivamente.

A Figura 35 apresenta conceitos relacionados ao HD-FCM II, voltado para o desvio de obstáculos. Em resumo, os pesos W14 e W35 são positivos, por outro lado, W34 e W15 são negativos. Esses valores são necessários para o desvio de obstáculos. Os pesos W24 e W25 conectados aos sensores frontais e as rodas têm valores negativos para promover uma desaceleração no robô quando o obstáculo está próximo.

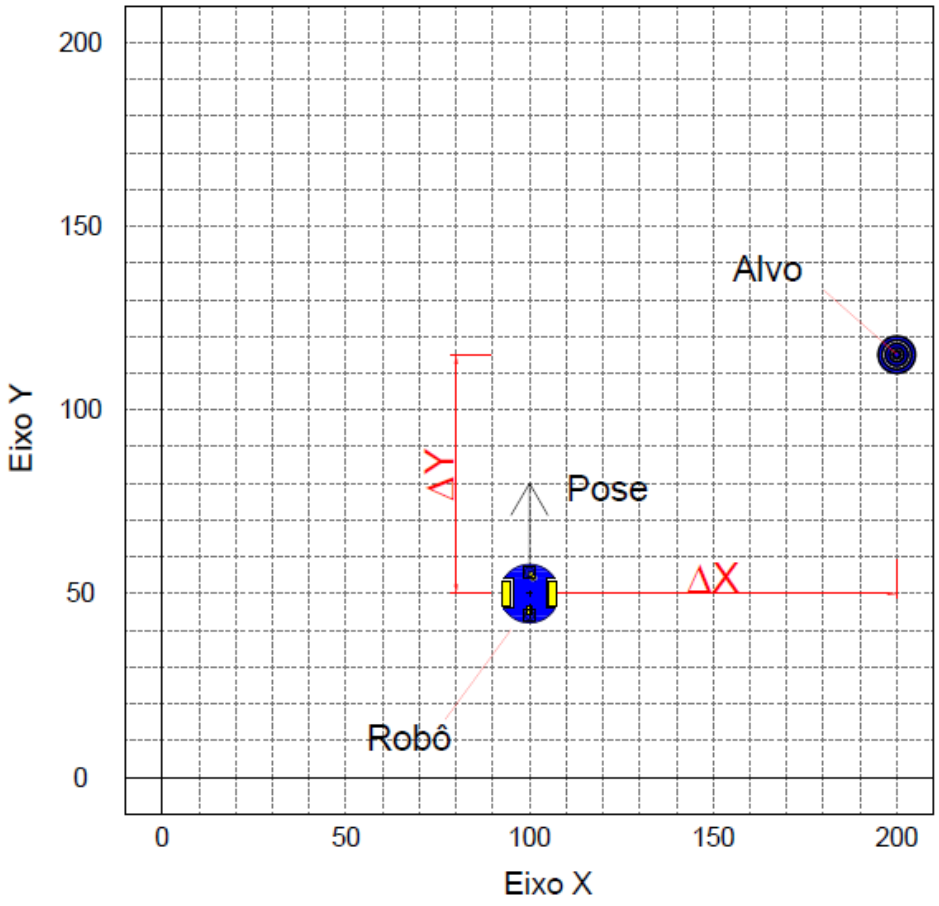


Figura 33 – Distâncias no cenário

Fonte: Autoria própria.

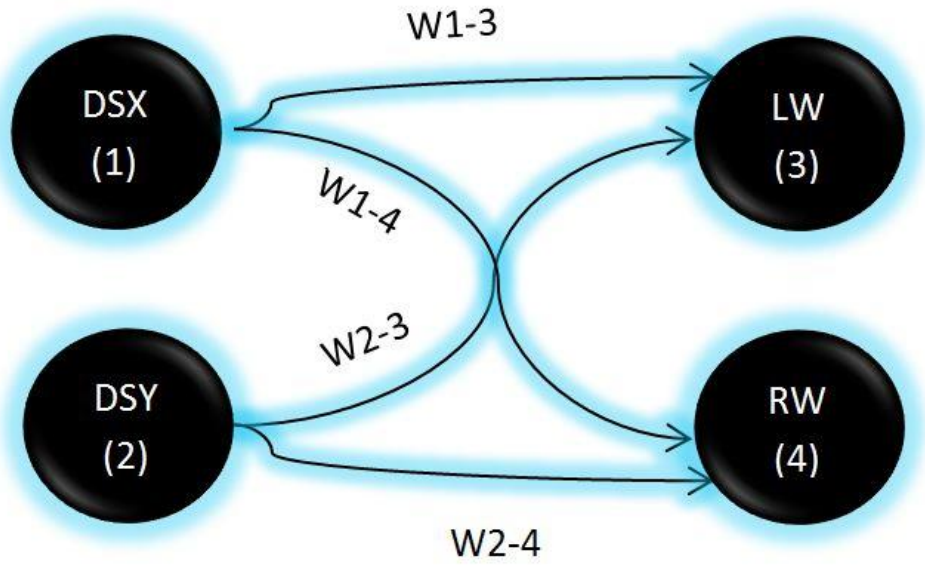


Figura 34 – HD-FCM I – Busca de alvo
Fonte: Autoria própria.

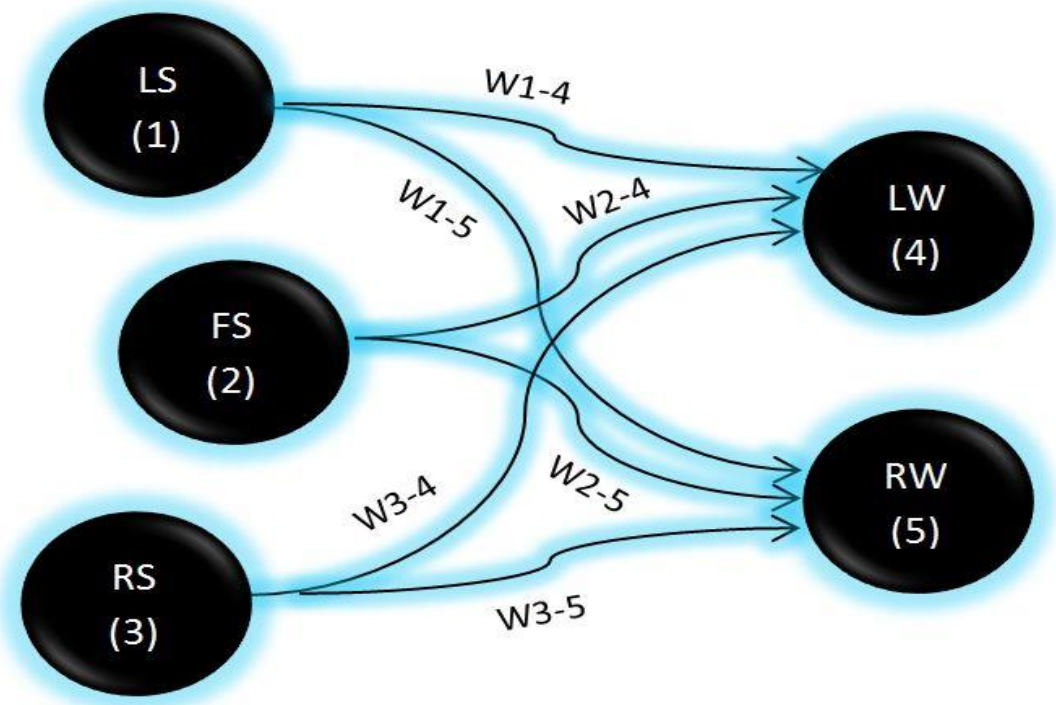


Figura 35 – HD-FCM II – Desvio de obstáculos
Fonte: Autoria própria.

O ajuste dos pesos *off-line* foi heurístico da seguinte forma: depois de determinado o sinal da relação casual, os valores dos pesos eram inicialmente de +0,5 ou -0,5 (dependendo da relação de causalidade entre os conceitos). O

próximo passo foi a partir da observação do comportamento do agente, ajustar os pesos até determinar os valores finais. O processo foi repetido até que atinja o alvo da melhor forma testada. E as matrizes de pesos encontradas foram:

FCM I

$$w' = \begin{vmatrix} 0 & 0 & 0{,}85 & -0{,}8 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

FCM II

$$w'' = \begin{vmatrix} 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0{,}1 & 0{,}1 \\ 0 & 0 & 0 & -0{,}9 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{vmatrix}$$

A leitura das matrizes é feita a partir do número da linha e coluna de cada termo, ou seja, $W'_{[1\ 3]}$ corresponde à influência que o elemento um possui sobre o elemento três. Estes pesos foram sintonizados baseando no o aprendizado de Hebb.

Para alternar entre o HD-FCM I e HD-FCM II foi utilizada uma máquina de estados finitos. A comutação é feita a partir da ocorrência de eventos. A priori o robô segue para o alvo, conforme há sinal de obstáculos nos sensores, ele inverte o HD-FCM passando a utilizar o FCM II voltado ao desvio de obstáculos. Não havendo sinais, volta para o estado inicial, considerando o HD-FCM I. O modelo da máquina de estados utilizado é apresentado na Figura 36.

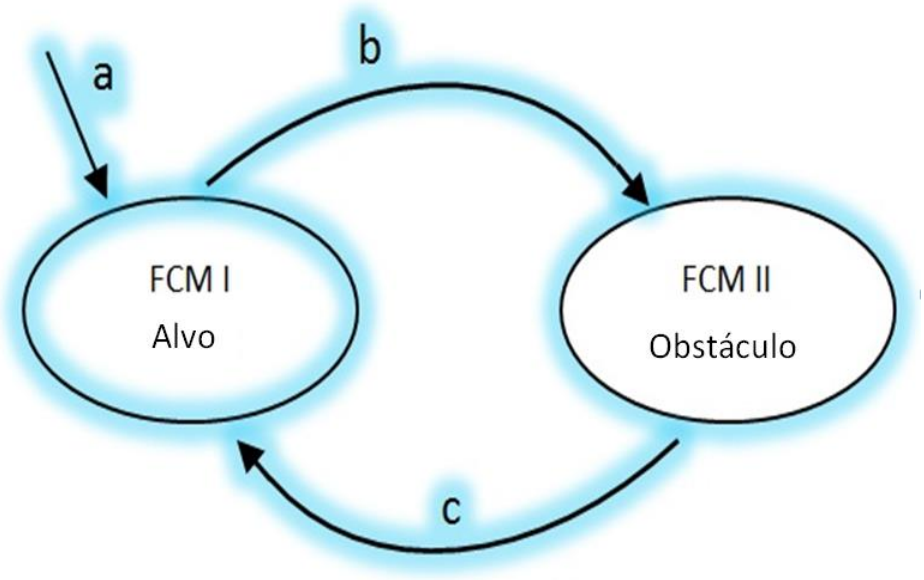


Figura 36 – Máquina de Estados Finitos

Fonte: Autoria própria.

O vocabulário da máquina utiliza os seguintes conceitos: a b e c da seguinte forma:

a) Inicia a máquina procurando o alvo.
b) Inverte o HD-FCM no caso de encontrar obstáculos.
c) Na ausência de obstáculos, retorna a buscar o alvo.

Na Seção 6.2, esta arquitetura será aplicada na navegação autônoma de um robô em ambiente simulado, e também em um experimento com o robô real.

5.4.2 Aprendizado de Hebb

Aprendizado de Hebb foi a primeira regra de aprendizado proposta por Donald Hebb em 1949 (HEBB, 1949). Aplicada inicialmente em Redes Neurais Artificiais, sua teoria é: “Se os dois neurônios, participantes de uma sinapse, têm ativação simultânea, então a força da conexão entre eles deve ser seletivamente aumentada” (HEBB, 1949) e baseando-se neste princípio, (STENT, 1973), (CHANGEUX; DANCHING, 1976) propuseram que se os neurônios são ativados de forma assíncrona, a sinapse deve ser seletivamente enfraquecida ou eliminada.

Para permitir simulações computacionais, essa regra recebeu algumas melhorias em 1956 (ROCHESTER *et al*., 1956) e foi expandida em 1988 (MCCLELLAND; RUMELHART, 1988), compondo assim a “regra de aprendizado expandida de Hebb” em que as unidades que são simultaneamente inibidas também reforçam os pesos em suas conexões.

A forma mais simples desta regra de aprendizado é descrita por (HAYKIN, 1999) na forma:

$$\Delta w_{kj}(n) = \eta y_k(n) x_j(n) \tag{58}$$

Onde: η é uma constante positiva que determina a taxa de aprendizado, x_j representa o sinal pré-sináptico, y_k representa o sinal pós-sináptico, w_{kj} representa o peso sináptico e **n**, a iteração do algoritmo de aprendizado.

A ideia desta regra está fortemente relacionada com o aprendizado por matriz de correlação proposto por Kohonen (1987) e Anderson (1972), que pode ser encontrado na literatura como algoritmo Hebbiano generalizado (AHG ou GHA – do inglês, *Generalized Hebbian Algorithm*).

Essa técnica foi utilizada neste trabalho para ampliar a capacidade de adaptação do agente móvel através da interação com o ambiente no sistema de navegação autônoma, fazendo o ajuste dos pesos dos HD-FCM's utilizados somando cada peso com um fator "n" multiplicado pela distância entre o robô e o alvo (dist). Deste modo, W[1-3] passou a ser representado pela expressão:

$$W[1-3] = W[1-3] + n\,.\,dist \tag{65}$$

Para todos os outros pesos foi feito de forma análoga.

6 RESULTADOS

Nas subseções apresentadas a seguir poderão ser observados os cenários de teste. Os mesmos testes foram feitos para as arquiteturas HWF e HD-FCM. A subseção 6.1 apresenta as simulações feitas para o HWF e 6.2 traz as simulações feitas para o HD-FCM, nas mesmas condições. Na subseção 6.3 é feita uma comparação entre as duas arquiteturas utilizadas.

Para cada cenário analisado, foram consideradas três simulações sendo organizados da seguinte forma:

a) cenário com obstáculos;

b) reorganização destes obstáculos;

c) mesmo cenário acrescentando a cada sensor ruídos randômicos de até cinco por cento do sinal captado nos sensores.

O ruído adicionado aos sensores objetiva gerar incertezas na aquisição de dados, característica de cenários com aquisições reais devido a problemas tais como o comportamento da propagação de ondas em diferentes materiais e ruídos fantasma. Em resumo é uma tentativa relativamente simples de aproximação de situações reais.

6.1 SIMULAÇÕES – SISTEMA HWF

Nesta subseção serão apresentadas as simulações feitas para o sistema HWF e os resultados obtidos, considerando número de passos para chegar ao alvo, precisão em relação ao alvo e posição final.

6.1.1 Cenário 01 – HWF

Neste cenário (Figuras 37 a 39) o protótipo se encontra na posição inicial (0,0) formando 0 grau com o eixo x, e o alvo localizado na posição (50,70). A representação dos obstáculos, alvos e trajetória são indicados na Figura 37. Todas as figuras seguem a mesma explicação.

A posição considerada do agente foi o centro de gravidade do robô. Desta forma é considerado alcance do alvo quando este ponto chega ao alvo com a precisão desejada, sendo essa indicada no início do simulador.

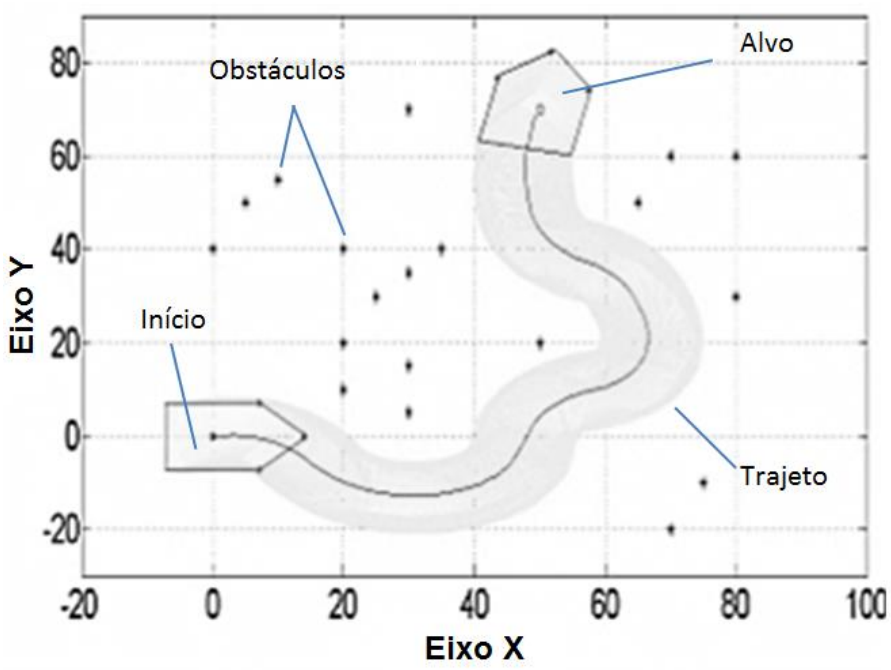


Figura 37 - Cenário 01 (a) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 361 passos (pulsos simultâneos nos dois motores) com uma precisão de 0,98 cm e a posição final do robô foi (49.34, 69.82).

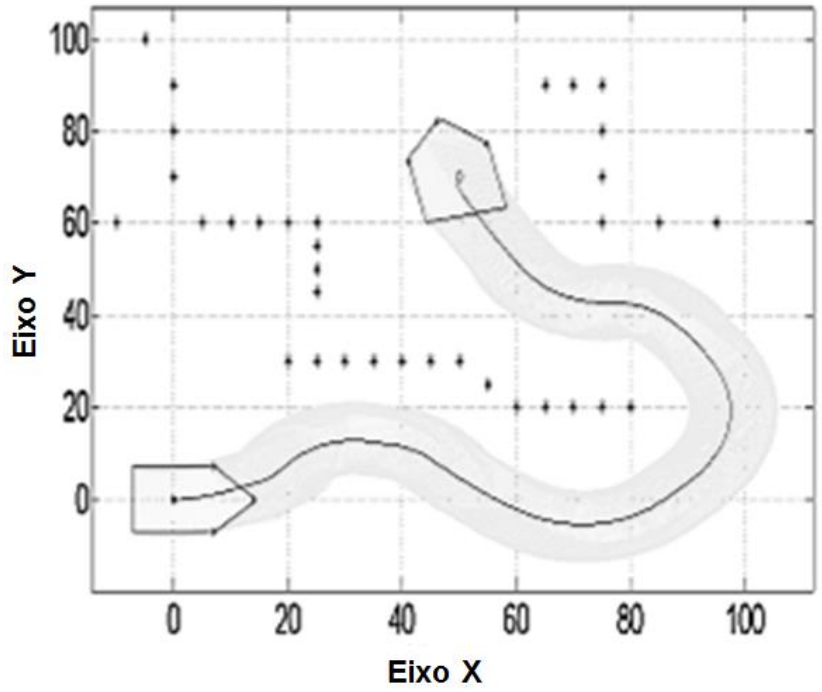


Figura 38 - Cenário 01 (b) HWF.

Fonte: Autoria própria.

Para esta simulação, o robô alcançou o alvo depois de 473 passos com uma precisão de 1,95 cm e a posição final do robô foi (48.66, 68.74).

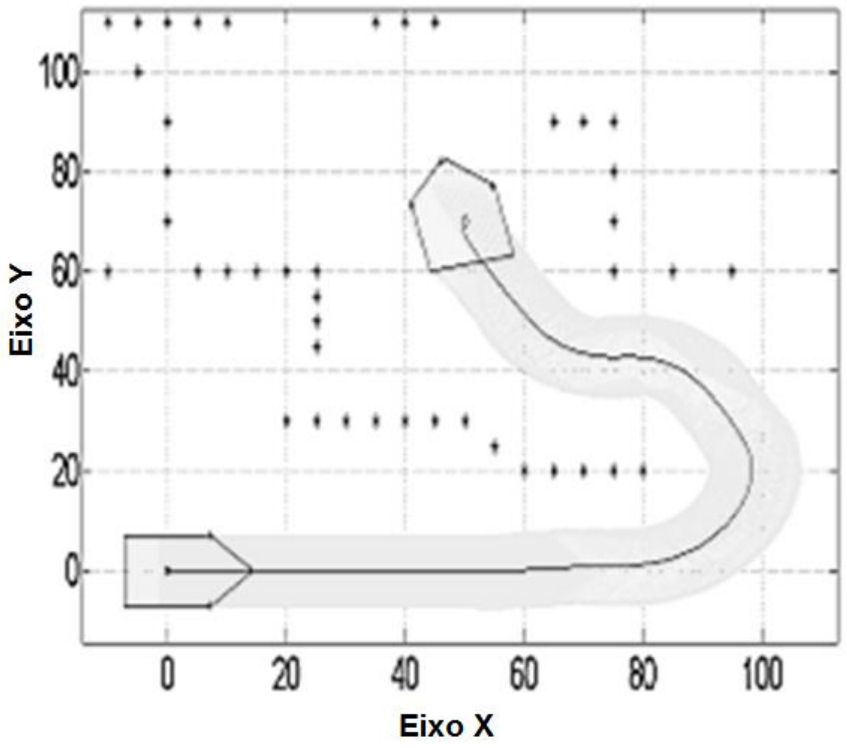


Figura 39 - Cenário 01 (c) HWF.

Fonte: Autoria própria.

Para esta simulação, o robô alcançou o alvo depois de 1051 passos com uma precisão de 1,93 cm e a posição final do robô foi (48.87, 68.65).

6.1.2 Cenário 02 – HWF

O protótipo neste caso se encontra na posição inicial (30,25) formando 45 graus com o eixo x e o alvo é localizado na posição (-50,80). As Figuras 40 a 42 apresentam a resposta a este segundo cenário.

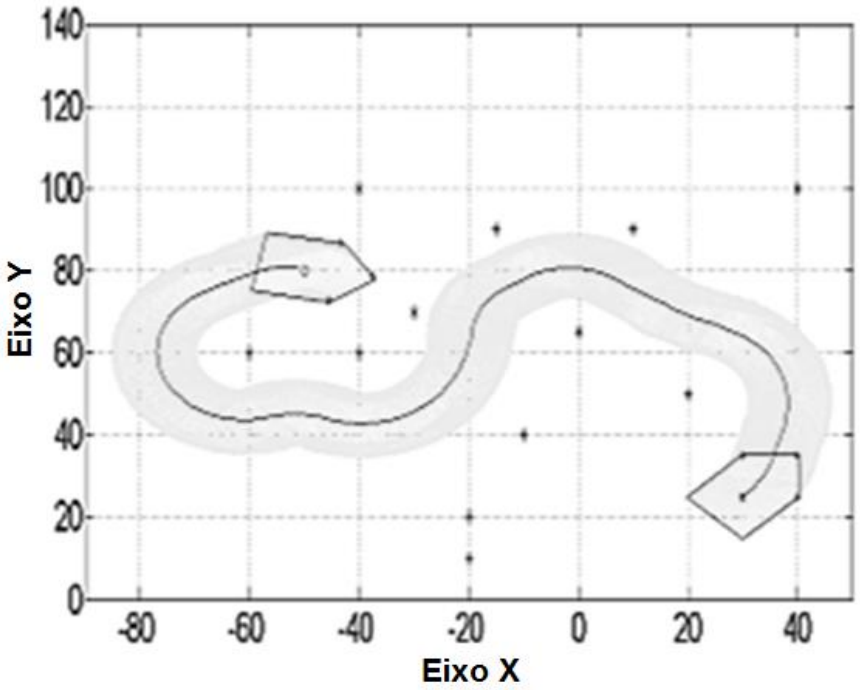


Figura 40 - Cenário 02 (a) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 549 passos com uma precisão de 0,82 cm e a posição final do robô foi (-50.22, 80.46).

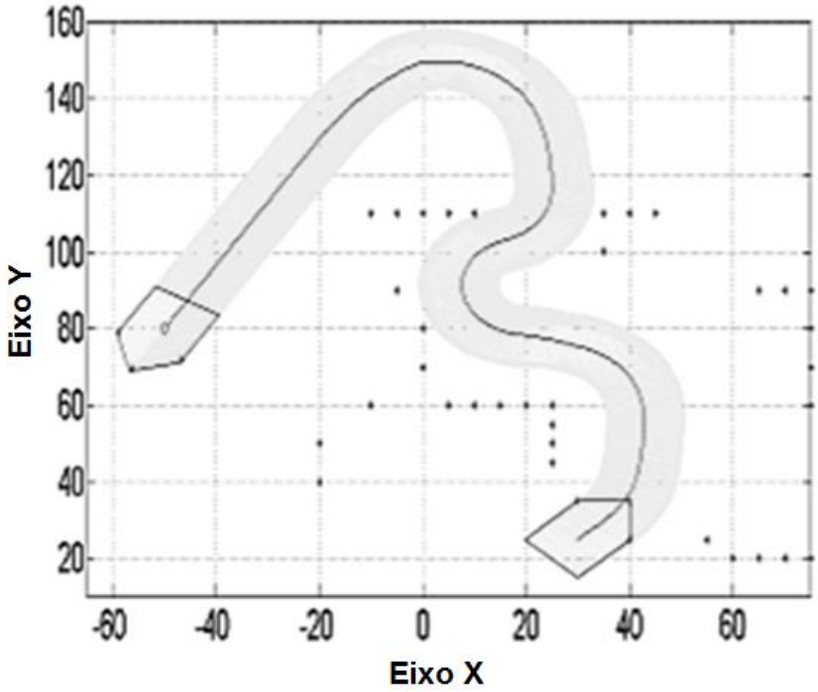
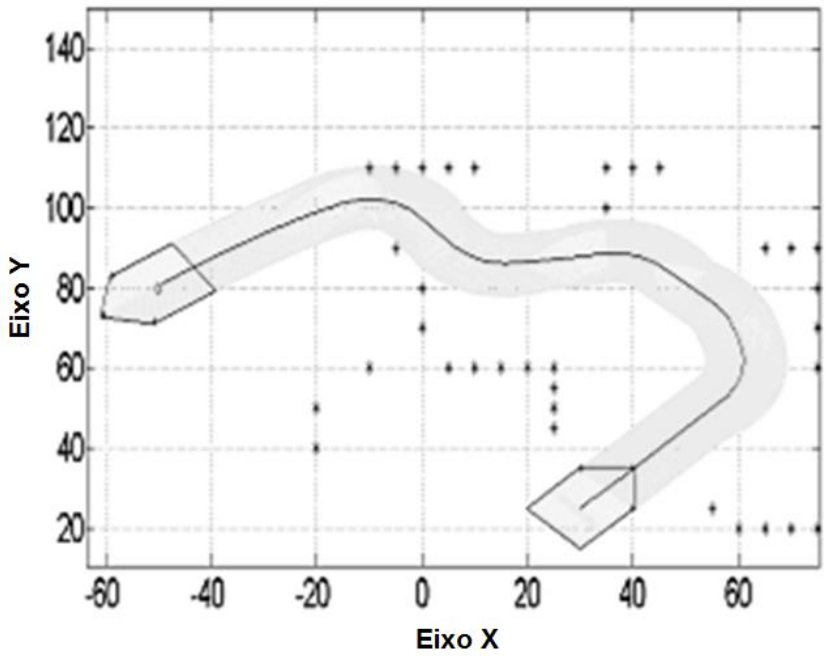


Figura 41 - Cenário 02 (b) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 655 passos com uma precisão de 1,71 cm e a posição final do robô foi (-49.32, 81.13).



(c)

Figura 42 - Cenário 02 (c) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 587 passos com uma precisão de 1,82 cm e a posição final do robô foi (-49.37, 81.05).

6.1.3 Cenário 03 – HWF

Na presente situação, o protótipo inicializa na posição (-80,75) formando 0 grau com o eixo x e alvo localizado na posição (35,75). As Figuras 43, 44 e 45 indicam a simulação referente a este cenário.

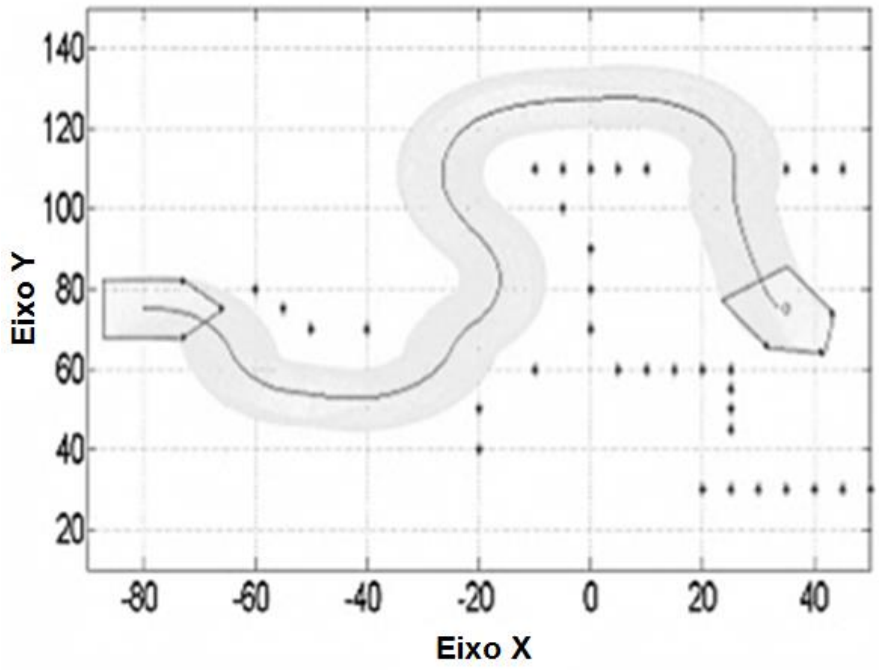


Figura 43 - Cenário 03 (a) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 583 passos com uma precisão de 1,92 cm e a posição final do robô foi (33.38, 74.27).

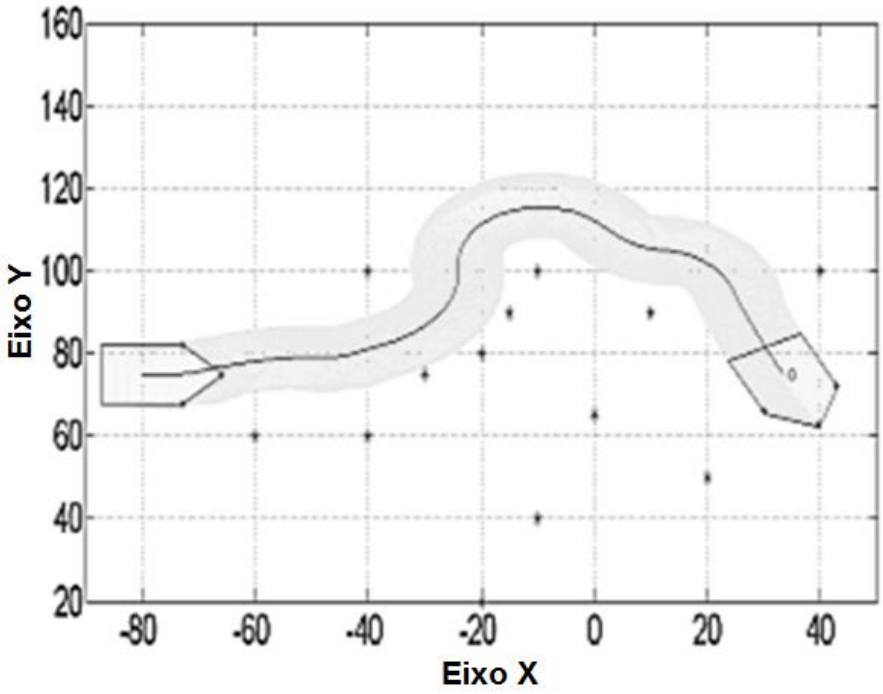


Figura 44 - Cenário 03 (b) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 397 passos com uma precisão de 1,43 cm e a posição final do robô foi (33.75, 74.34).

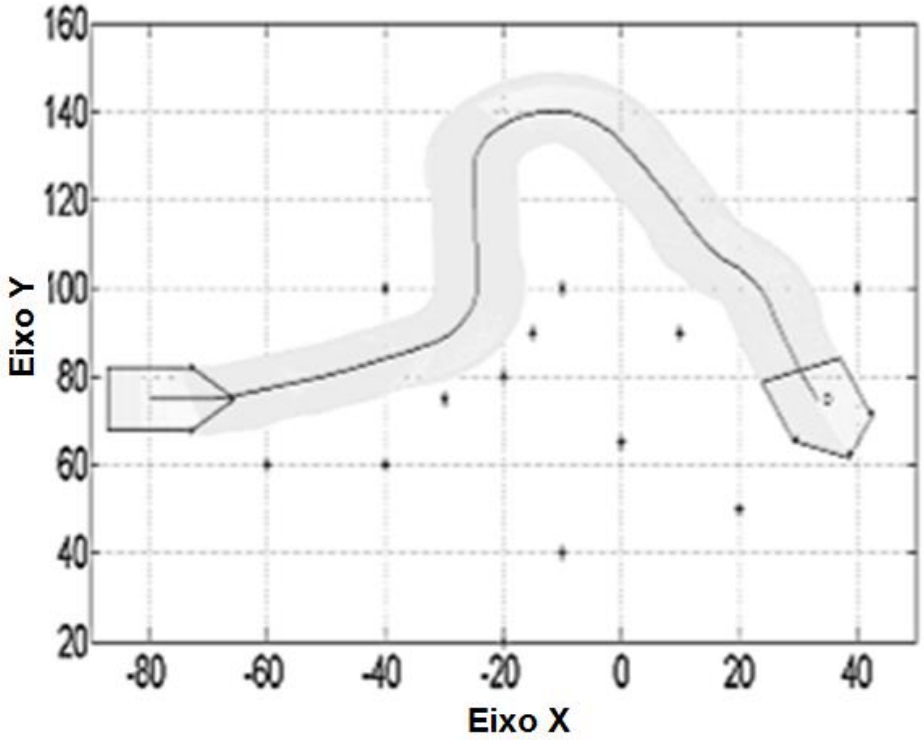


Figura 45 - Cenário 03 (c) HWF.

Fonte: Autoria própria.

No cenário 3 com ruído, o robô alcançou o alvo depois de 974 passos com uma precisão de 1,96 cm e a posição final do robô foi (33.18, 74.42).

6.1.4 Cenário 04 – HWF

O protótipo neste caso se encontra na posição inicial (-50, -90) formando 0 grau com o eixo x e o alvo é localizado na posição (0,0). Esta simulação foi feita considerando a complexidade de um cenário em espiral, conforme apresentado na Figura 46 e 47. Neste cenário não foi considerada a situação (b) descrita anteriormente (referente à reorganização dos obstáculos).

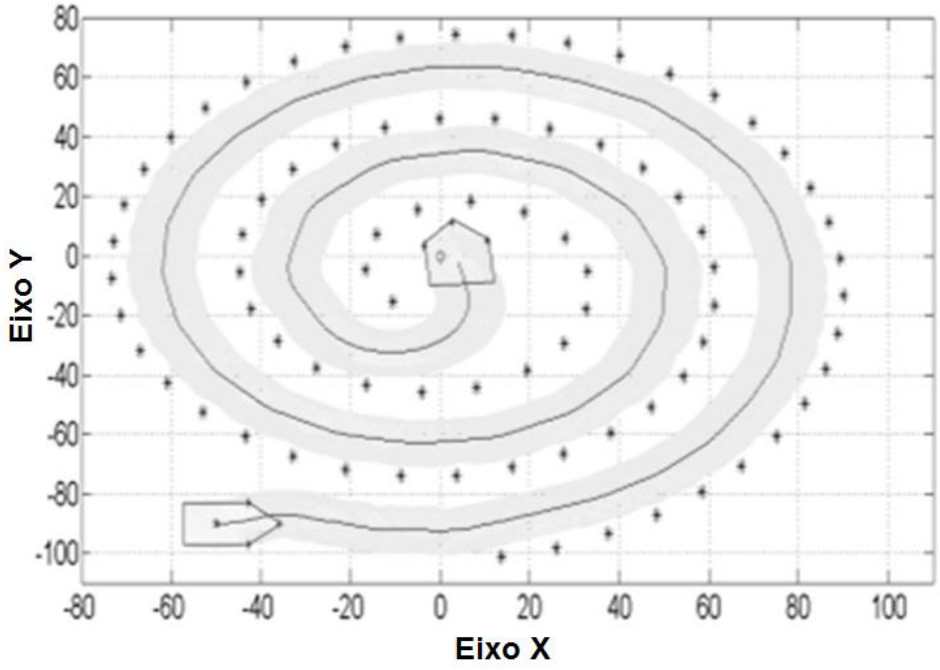


Figura 46 - Cenário 04 (a) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 1960 passos com uma precisão de 9,67 cm e a posição final do robô foi (4.43, -8.85).

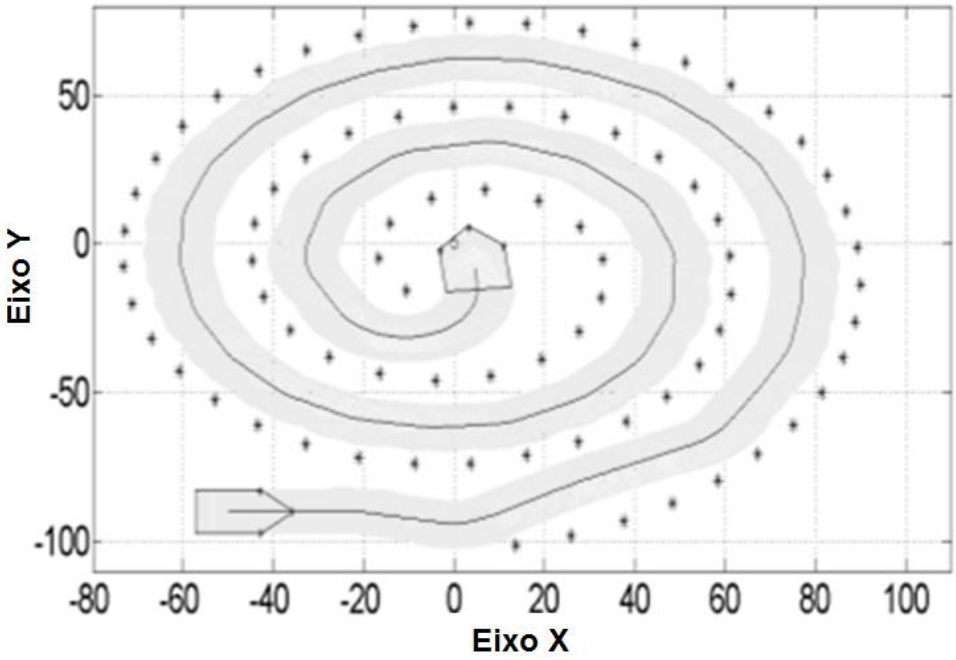


Figura 47 - Cenário 04 (b) HWF.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 2421 passos com uma precisão de 9,93 cm e a posição final do robô foi (4.60, -8.38).

Para o cenário em espiral, considerando o fato de ter um obstáculo muito próximo do alvo (menos de 15 cm), torna-se necessário reajustar o erro desejado para 10 cm. Caso contrário o sistema daria prioridade para o desvio de obstáculos e nunca chegaria ao alvo, ficando somente desviando dos obstáculos.

6.1.5 Cenário 05 – Dois Alvos – HWF

Como uma evolução do trabalho prévio, no objetivo de fazer a gerenciamento e manutenção de energia de um robô real, por exemplo, recarga das baterias, foi desenvolvida uma estratégia em que o agente busque dois alvos ou dois pontos de recarga. Desse modo, O agente busca inicialmente o alvo de número um (xf_1, yf_1) e posteriormente, o alvo de número dois (*xf2*, *yf2*).

A função de que o robô atinja mais de um alvo serve também para o caso em que ele precise em uma indústria, por exemplo, pegar alguma peça de um lugar (alvo 01) e transportá-la para outro (alvo 02).

Nesta estratégia, o algoritmo inicial (busca de um alvo) é rodado duas vezes, no caso de dois alvos e assim por diante. O primeiro *loop* utiliza como referência, a distância do robô e o primeiro alvo, e o segundo *loop*, utiliza como referência a distância entre o robô e o segundo alvo. Resumindo, o ponto inicial do segundo *loop* de procura ao alvo refere-se ao ponto final do primeiro loop.

Neste cenário, foram feitas quatro simulações considerando dois alvos dispostos em localidades diferentes para cada simulação. A posição inicial é (-80,75) formando zero grau com o eixo x. Cada caso será apresentado nas Figuras 48 a 52.

Para o caso “a”, os alvos se encontravam na posição (35,75) e (80, -20), respectivamente. O erro desejado foi de 5 centímetros.

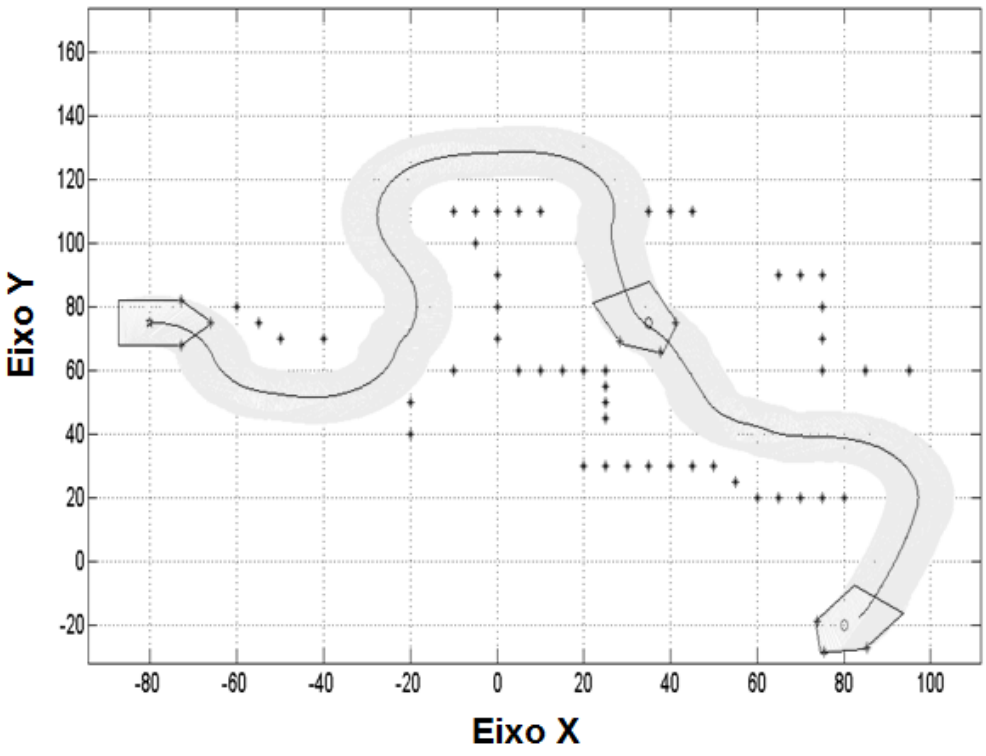


Figura 48 - Cenário 05 (a) HWF.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 1,68 cm depois de 936 passos. A posição ao atingir o primeiro alvo foi (33.52, 75.49), e (81.24, -19.63), a posição final.

No caso “b”, houve uma modificação no cenário, sendo acrescentado um obstáculo na posição (20,110) forçando o agente a tomar uma nova rota.

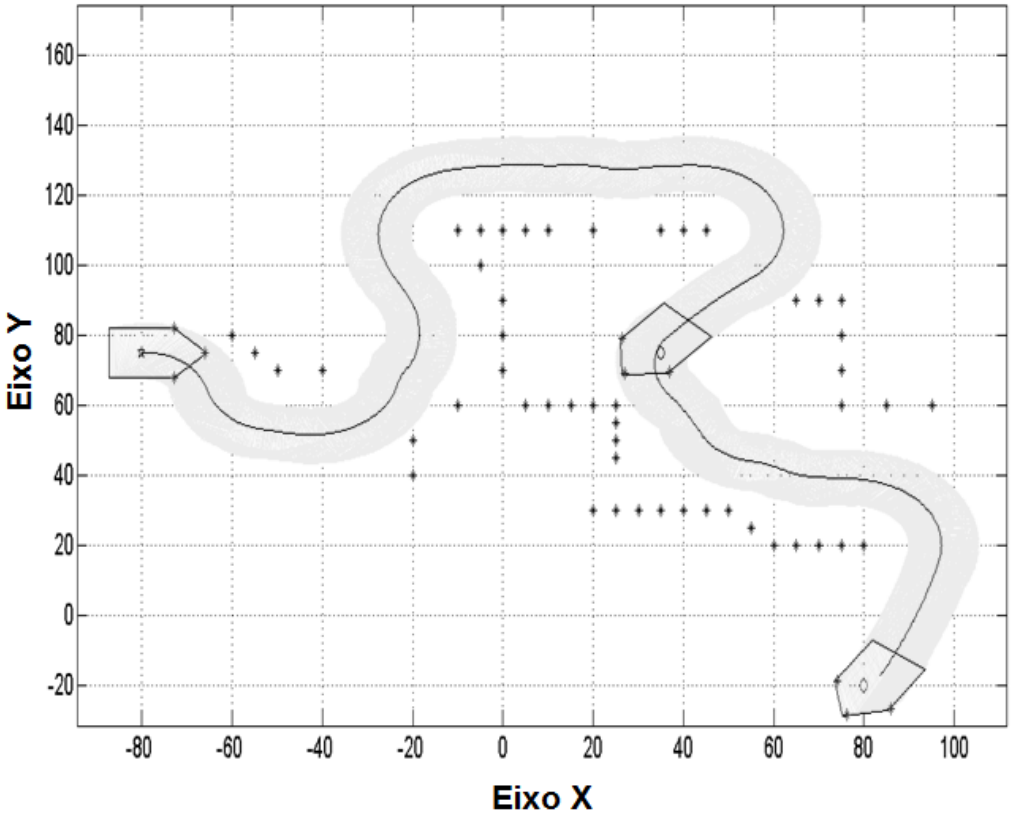


Figura 49 - Cenário 05 (b) HWF.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 5 cm depois de 1047 passos. A posição ao atingir o primeiro alvo foi (36.14, 79.20), e (83.58, -17,00), a posição final.

No caso “c” foi feito a mesma simulação do caso anterior, mas o erro desejado foi reduzido para 1 centímetro.

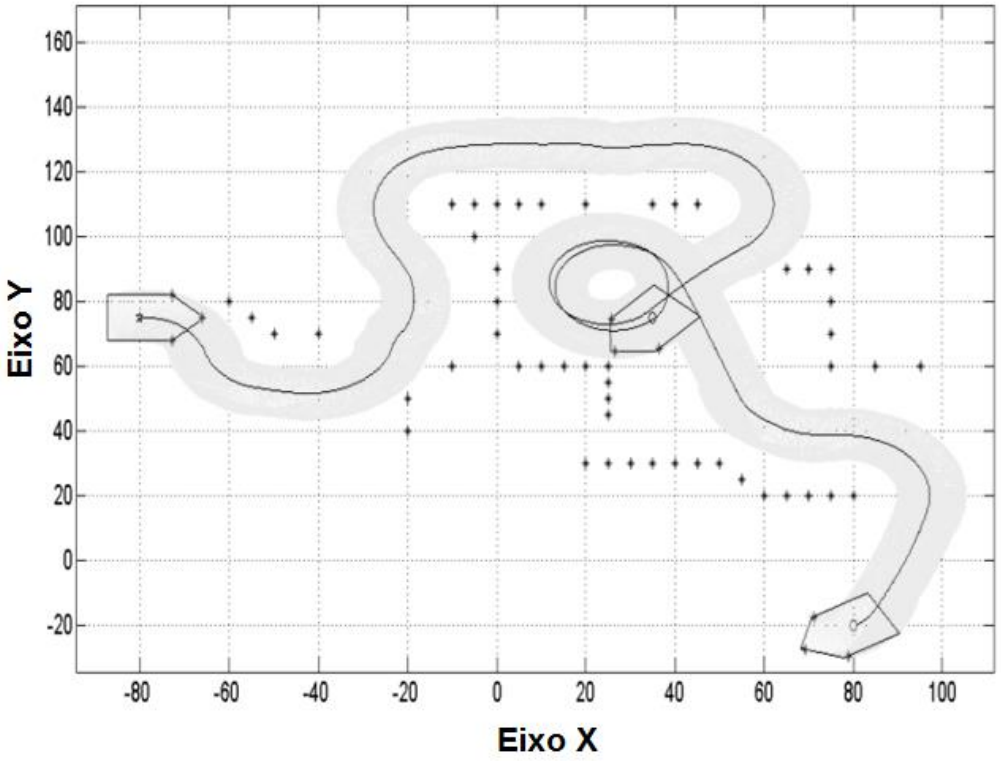


Figura 50 - Cenário 05 (c) HWF.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 0,97 cm depois de 1455 passos. A posição ao atingir o primeiro alvo foi (35.50, 74.97), e (80.62, -20.00), a posição final.

Note que o robô atingiu os objetivos com a precisão desejada, porém teve que fazer duas voltas (em torno da posição (30,80)) para alcançar o primeiro objetivo, pois ainda priorizava o desvio de obstáculos por não ter atingido a precisão desejada na primeira passada. Um exemplo do caso no qual o aumento na precisão pode ocasionar também um aumento desnecessário na trajetória.

Para o caso “d”, houve uma alteração na posição do segundo obstáculo, localizando o mesmo na posição (80,140).

O robô alcançou os alvos com uma precisão de 4,84 cm depois de 1022 passos. A posição ao atingir o primeiro alvo foi (36.15, 79.20), e (78.62, 135.76), a posição final.

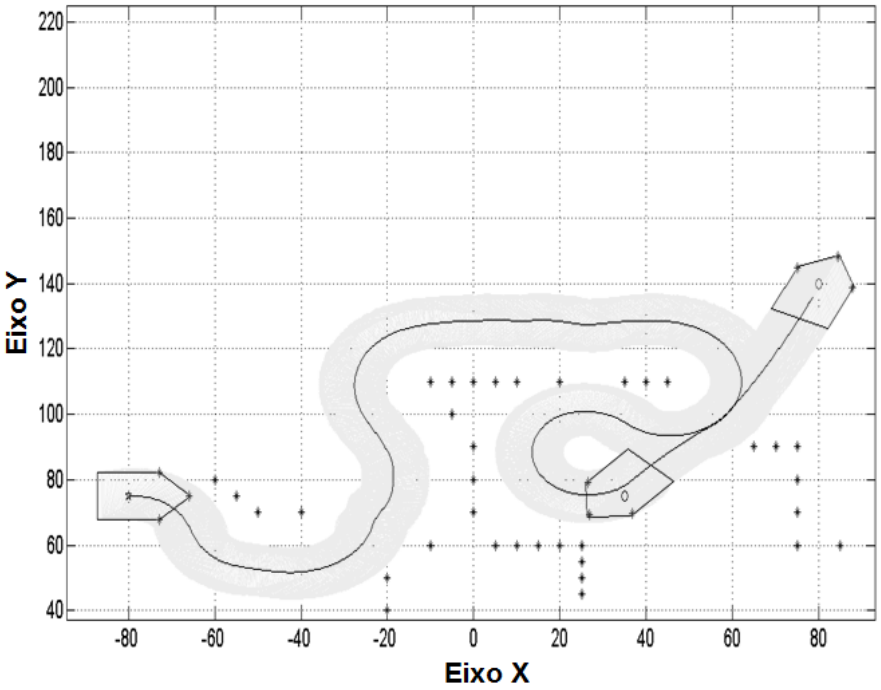


Figura 51 - Cenário 05 (d) HWF.

Fonte: Autoria própria.

Por fim, foi invertida a ordem dos alvos do caso “e”, fazendo com que o robô busque primeiro a posição (80, -20) e posteriormente siga para o alvo final (35,75).

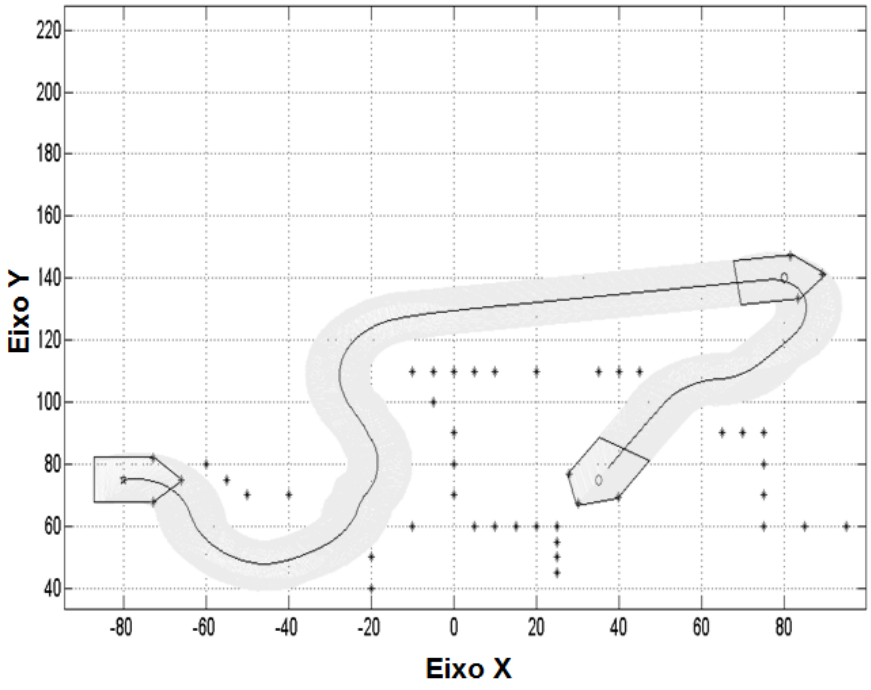


Figura 52 - Cenário 05 (e) HWF.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 0,15 cm depois de 871 passos. A posição ao atingir o primeiro alvo foi (80.45, 140.05), e (34.68, 74.55), a posição final.

6.1.6 Cenário 06 – Aumento No Erro – HWF

Essa simulação foi feita para demonstrar um caso específico: o alvo está localizado muito perto de um obstáculo, então para este caso, o robô chegou ao alvo com um erro mínimo de 8 cm, se mantivéssemos o erro mínimo de 1cm, ele ficaria dando volta em torno dos obstáculos em formato de “L”.

Para o caso em que o alvo se encontra em uma posição impossível, dentro de um círculo de obstáculos, por exemplo, ou dentro de uma sala fechada, ele tentaria um número de passos de mil vezes a distância em linha reta da posição inicial até a final. Caso não chegue ao alvo, imprimirá uma mensagem: Alvo impossível.

Para o caso apresentado, a posição inicial do robô é (0,0) formando 0 grau com o eixo x, e a posição final (80,50). A distância em linha reta inicial era de 94,34 cm. A Figura 53 apresenta a simulação do caso descrito.

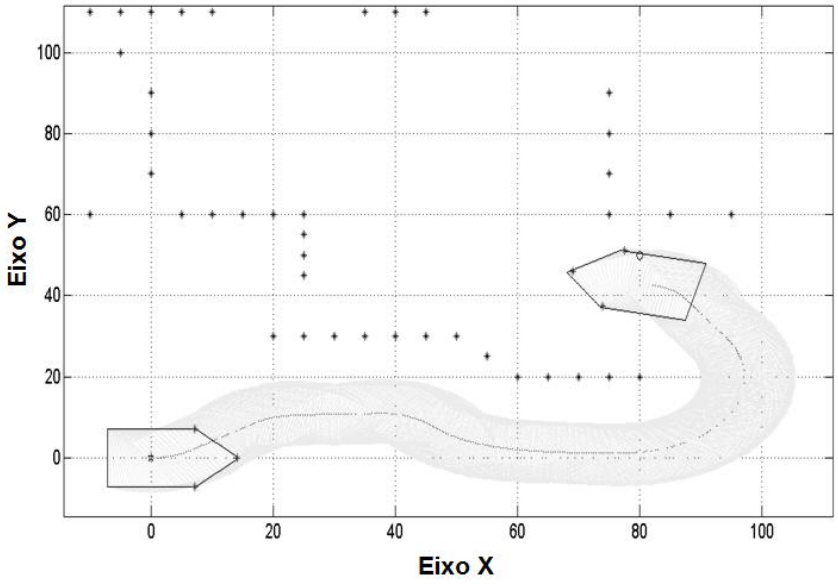


Figura 53 - Cenário 06 (a) HWF.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 7,94 cm depois de 360 passos. A posição ao atingir o alvo foi (82.12, 42.56).

6.2 SIMULAÇÕES – CONTROLADOR HD-FCM

Nesta subseção as mesmas simulações serão repetidas, também considerando número de passos para chegar ao alvo, precisão em relação ao alvo e posição final, entretanto o sistema de controle utilizado para o alcance de alvo e desvio de obstáculos foi a HD-FCM, conforme apresentado na subseção 5.4. Isso é importante, pois para se comparar duas técnicas é necessário que sejam feitas nas mesmas condições (MENDONÇA, 2011).

6.2.1 Cenário 01 – HD-FCM

Neste cenário (Figuras 54 a 56) o protótipo se encontra na posição inicial (0,0) formando 0 graus com o eixo x, e o alvo localizado na posição (50,70).

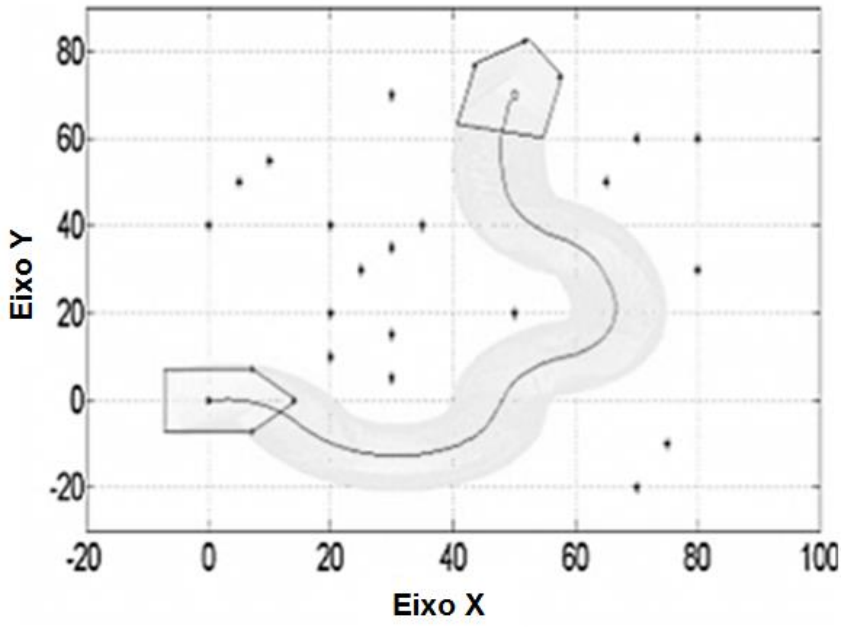


Figura 54 - Cenário 01 (a) HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 358 passos (pulsos simultâneos nos dois motores) com uma precisão de 0,97 cm e a posição final do robô foi (49.36, 69.84).

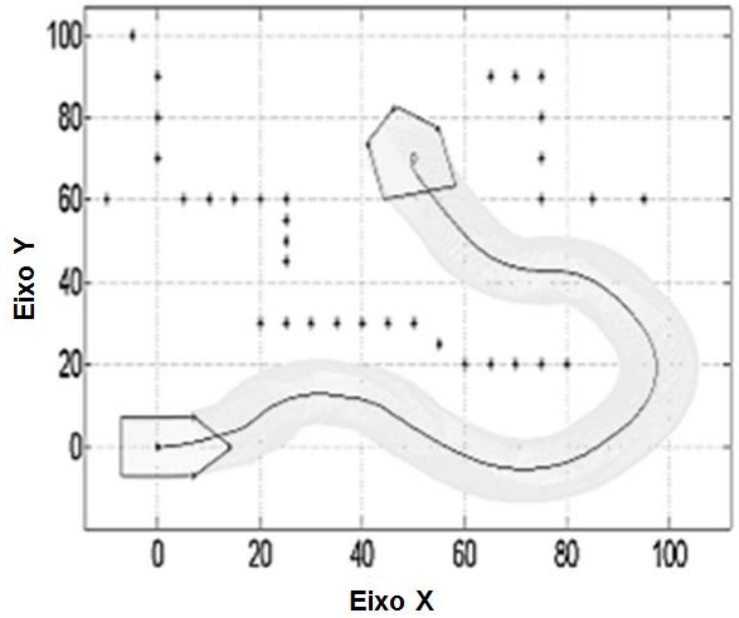


Figura 55 - Cenário 01 (b) HD-FCM.

Fonte: Autoria própria.

Para esta simulação, o robô alcançou o alvo depois de 473 passos com uma precisão de 1,94 cm e a posição final do robô foi (48.67, 68.74).

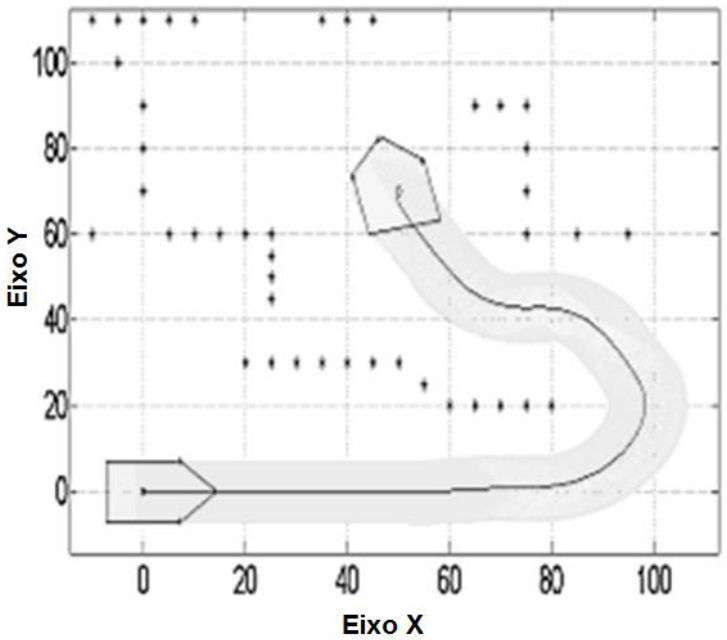


Figura 56 - Cenário 01 (c) HD-FCM.

Fonte: Autoria própria.

Para esta simulação, o robô alcançou o alvo depois de 1048 passos com uma precisão de 1,93 cm e a posição final do robô foi (48.88, 68.64).

6.2.2 Cenário 02 – HD-FCM

O protótipo neste caso se encontra na posição inicial (30,25) formando 45 graus com o eixo x e o alvo é localizado na posição (-50,80). As Figuras 57, 58 e 59 apresentam as respostas para este cenário.

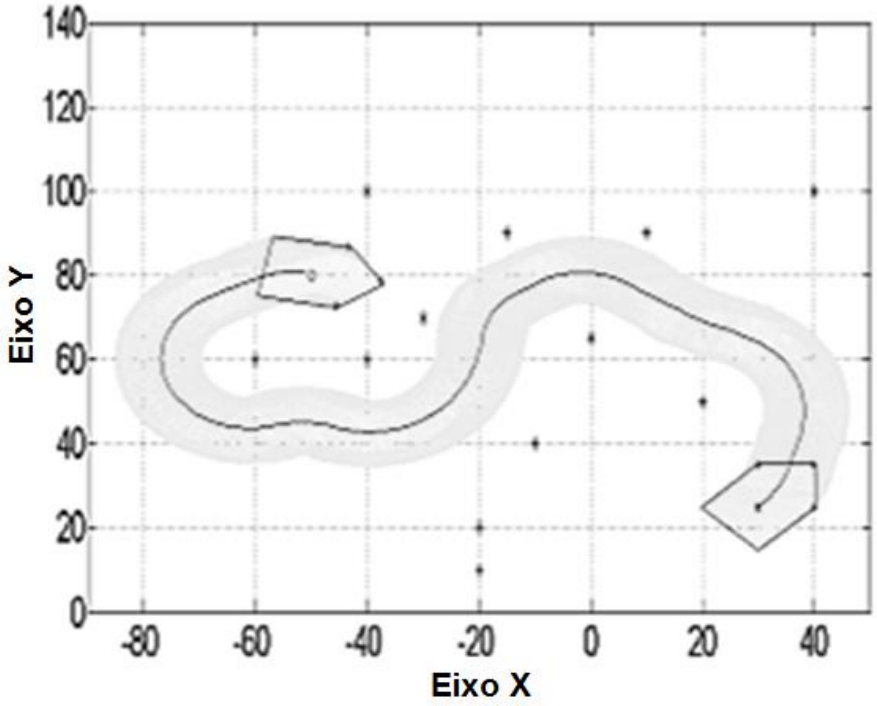


Figura 57 - Cenário 02 (a) HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 545 passos com uma precisão de 0,81 cm e a posição final do robô foi (-50.23, 80.48).

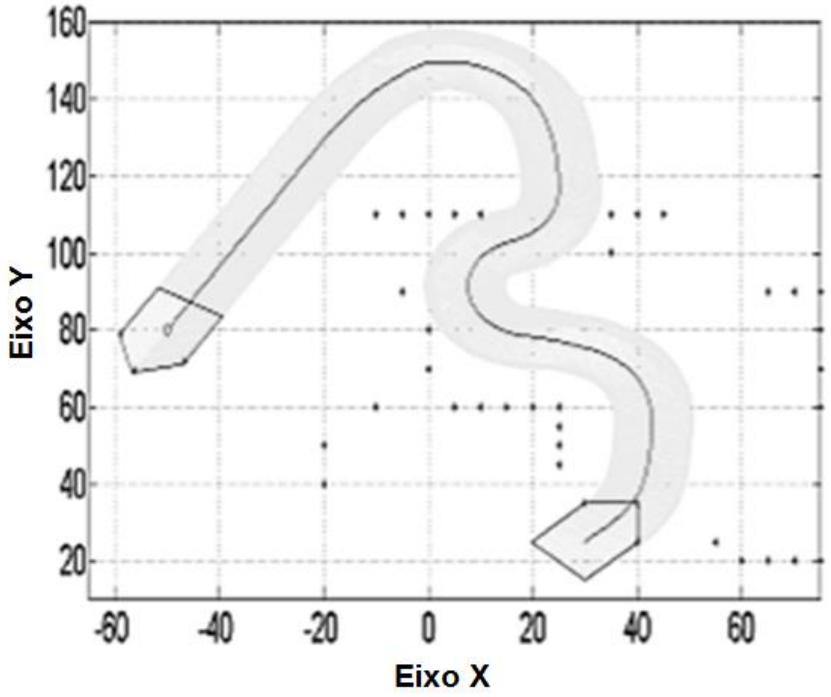


Figura 58 - Cenário 02 (b) HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 651 passos com uma precisão de 1,69 cm e a posição final do robô foi (-49.35, 81.12).

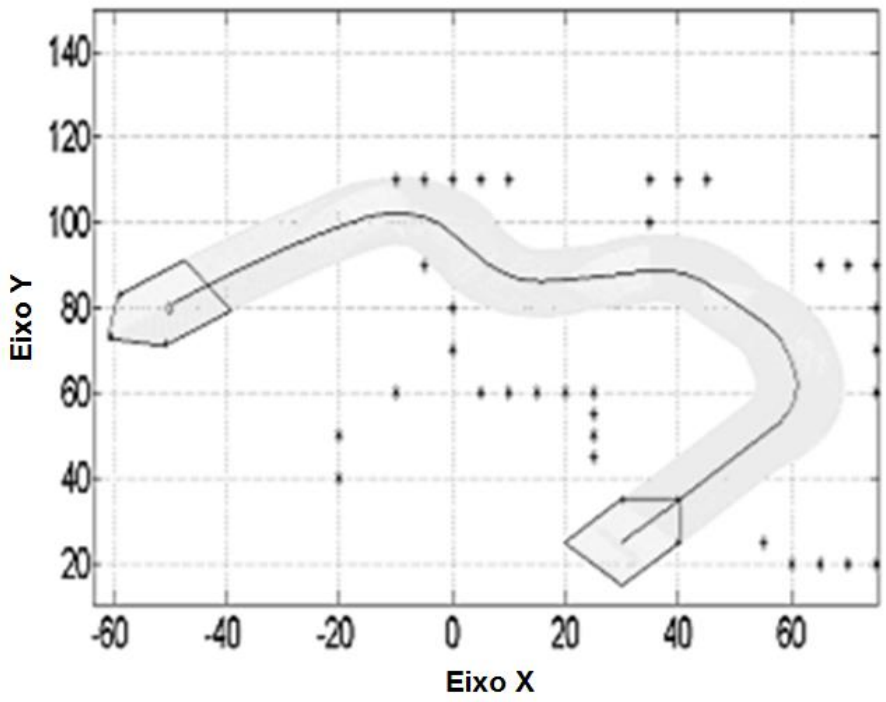


Figura 59 - Cenário 02 (c) HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 584 passos com uma precisão de 1,82 cm e a posição final do robô foi (-49.38, 81.06).

6.2.3 Cenário 03 – HD-FCM

Neste cenário, o protótipo inicializa na posição (-80,75) formando 0 grau com o eixo x e alvo localizado na posição (35,75). As Figuras 60 a 62 indicam as simulações referente a este cenário.

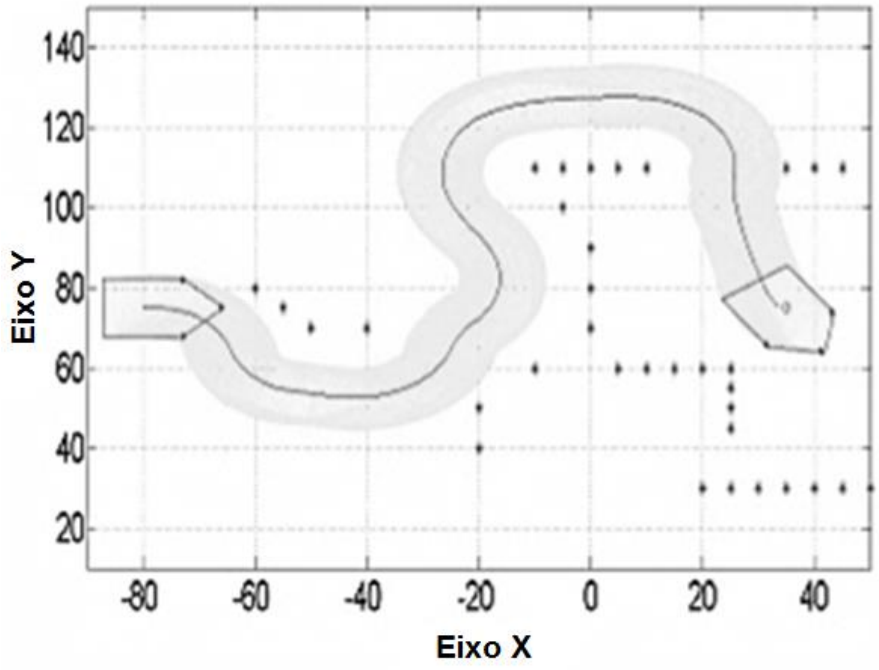


Figura 60 - Cenário 03 (a) HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 579 passos com uma precisão de 1,93 cm e a posição final do robô foi (33.37, 74.29).

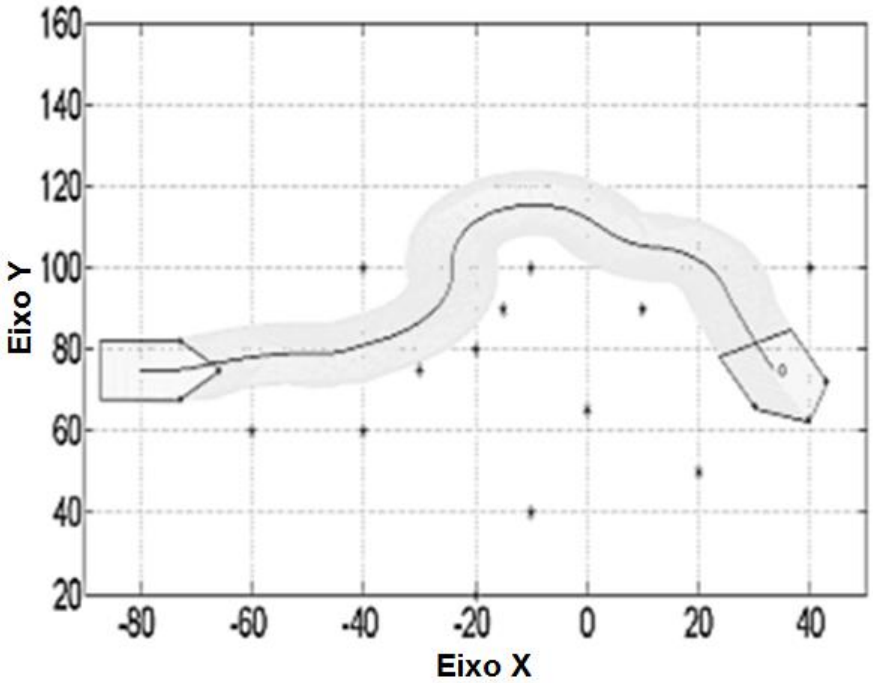


Figura 61 - Cenário 03 (b) HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 393 passos com uma precisão de 1,42 cm e a posição final do robô foi (33.77, 74.33).

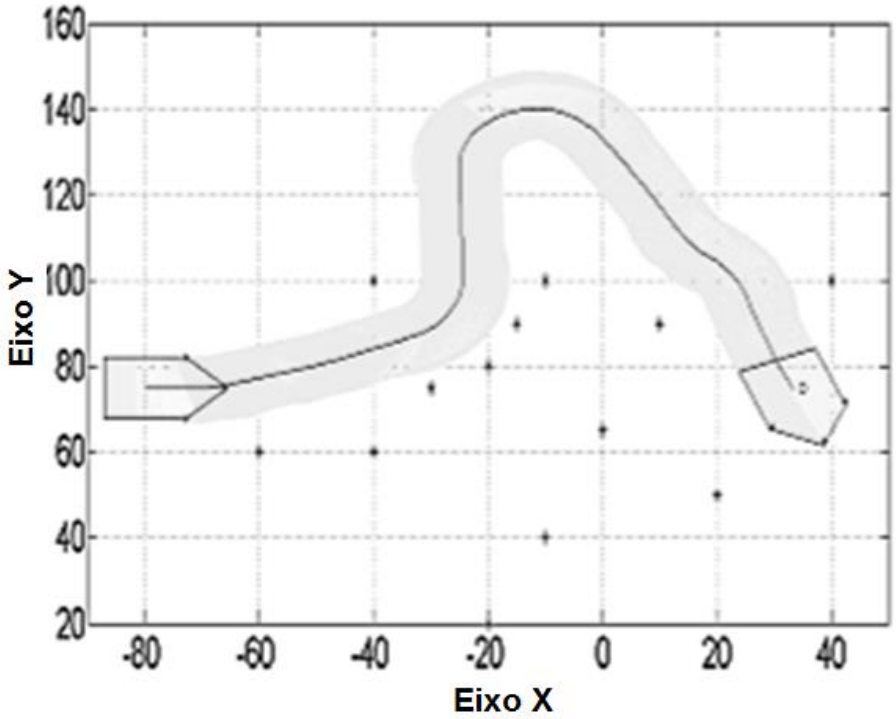


Figura 62 - Cenário 03 (c) HD-FCM.

Fonte: Autoria própria.

No cenário 03 com ruído, o robô alcançou o alvo depois de 980 passos com uma precisão de 1,97 cm e a posição final do robô foi (33.17, 74.43).

6.2.4 Cenário 04 – HD-FCM

O protótipo neste caso se encontra na posição inicial (-50, -90) formando 0 grau com o eixo x e o alvo é localizado na posição (0,0). Esta simulação foi feita considerando a complexidade de um cenário em espiral, conforme apresentado nas Figuras 63 e 64. Neste cenário não foi considerado a situação (b) descrita anteriormente (referente à reorganização dos obstáculos).

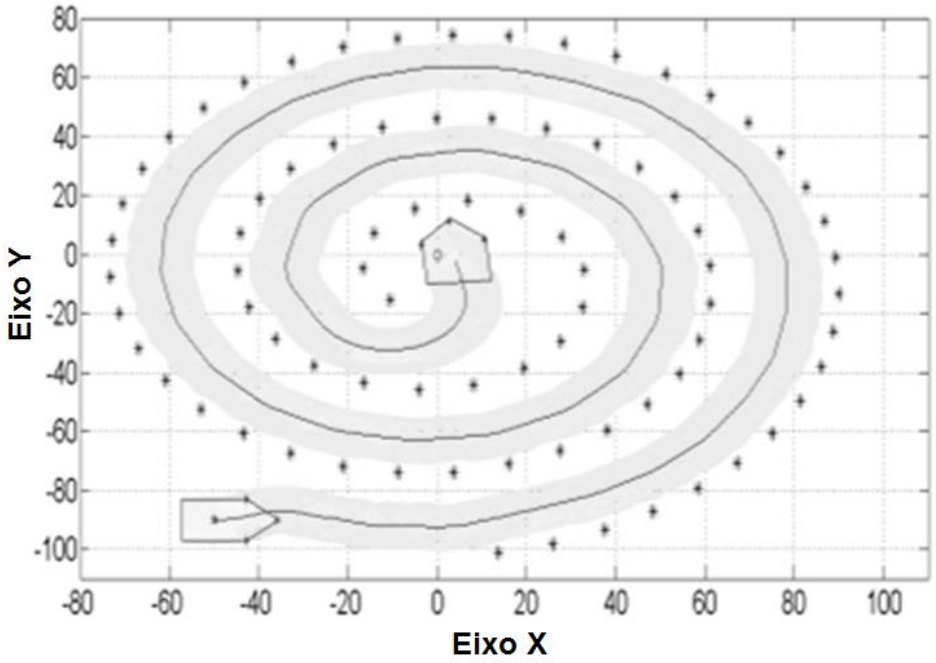


Figura 63 - Cenário 04 (a) - HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 1962 passos com uma precisão de 9,67 cm e a posição final do robô foi (4.43, -8.85).

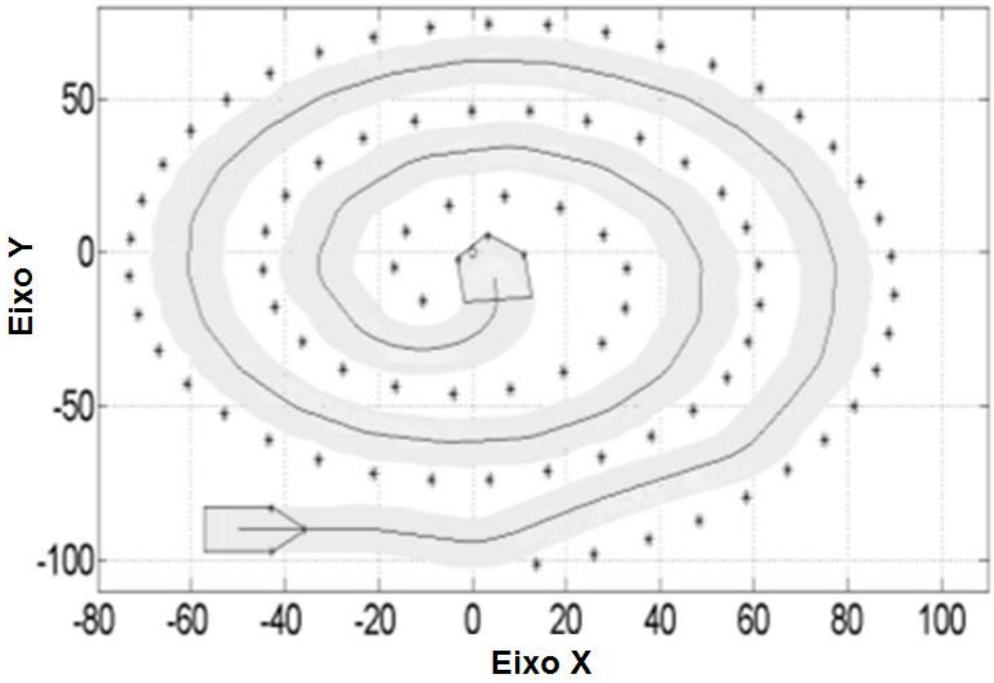


Figura 64 - Cenário 04 (b) HD-FCM.

Fonte: Autoria própria.

O robô alcançou o alvo depois de 2419 passos com uma precisão de 9,94 cm e a posição final do robô foi (4.61, -8.37).

6.2.5 Cenário 05 – Dois Alvos – HD-FCM

Neste cenário, foram feitas quatro simulações considerando dois alvos dispostos em localidades diferentes para cada simulação. A posição inicial é (-80,75) formando zero grau com o eixo x. Cada caso será apresentado nas Figuras 65 a 69.

Para o caso “a”, os alvos se encontravam na posição (35,75) e (80, -20), respectivamente. O erro desejado foi de 05 centímetros.

O robô alcançou os alvos com uma precisão de 1,68 cm depois de 935 passos. A posição ao atingir o primeiro alvo foi (33.53, 75.49), e (81.24, -19.64), a posição final.

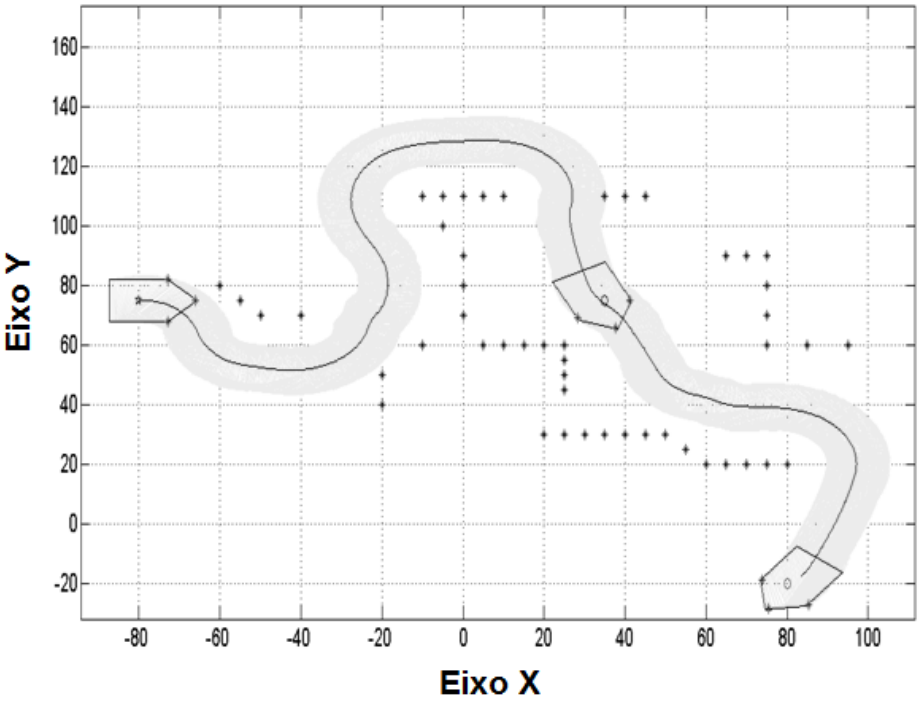


Figura 65 - Cenário 05 (a) HD-FCM.

Fonte: Autoria própria.

No caso “b”, houve uma modificação no cenário, sendo acrescentado um obstáculo na posição (20,110) forçando o agente a tomar uma nova rota.

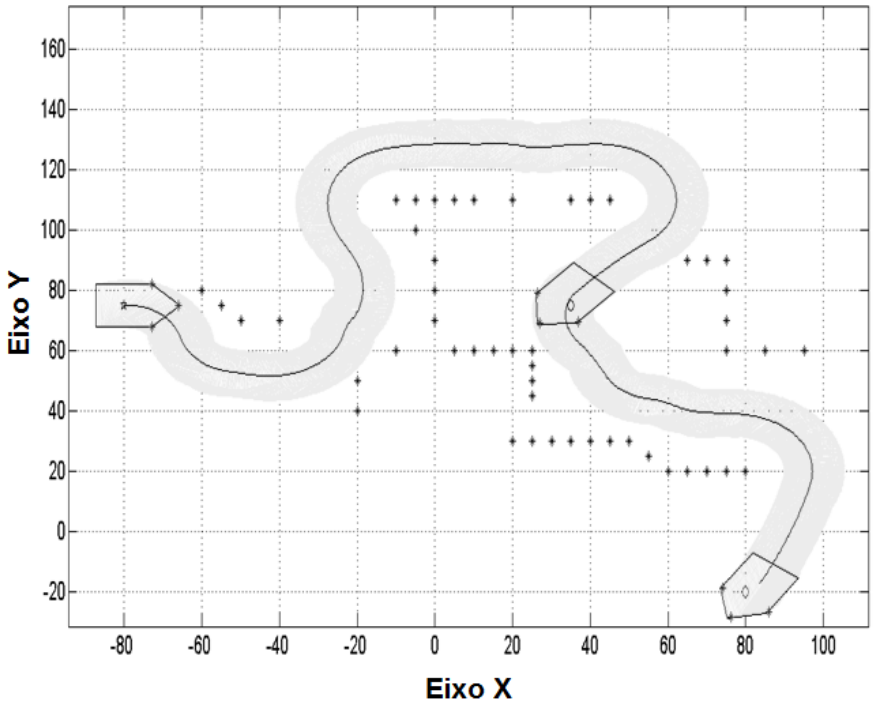


Figura 66 - Cenário 05 (b) - HD-FCM.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 05 cm depois de 1044 passos. A posição ao atingir o primeiro alvo foi (36.15, 79.20), e (83.58, -17,00), a posição final.

No caso “c” foi feito a mesma simulação do caso anterior, mas o erro desejado foi reduzido para 01 centímetro.

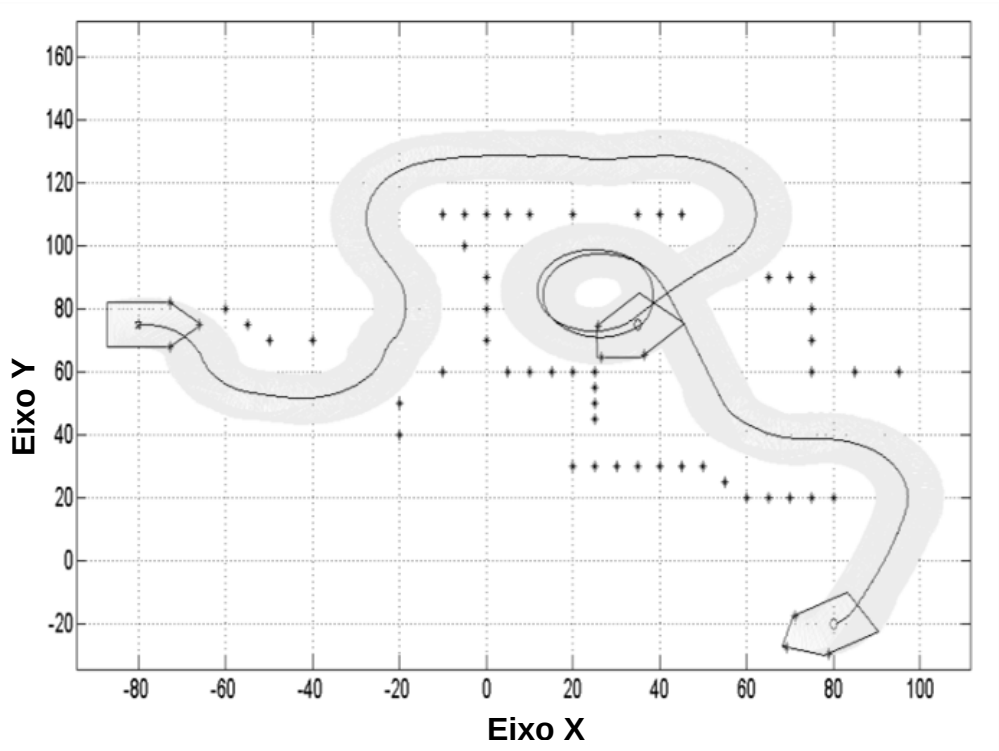


Figura 67 - Cenário 05 (c) HD-FCM.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 0,95 cm depois de 1458 passos. A posição ao atingir o primeiro alvo foi (35.52, 74.96), e (80.61, -20.01), a posição final.

Para o caso “d”, houve uma alteração na posição do segundo obstáculo, localizando o mesmo na posição (80,140).

O robô alcançou os alvos com uma precisão de 4,83 cm depois de 1024 passos. A posição ao atingir o primeiro alvo foi (36.17, 79.24), e (78.64, 135.77), a posição final.

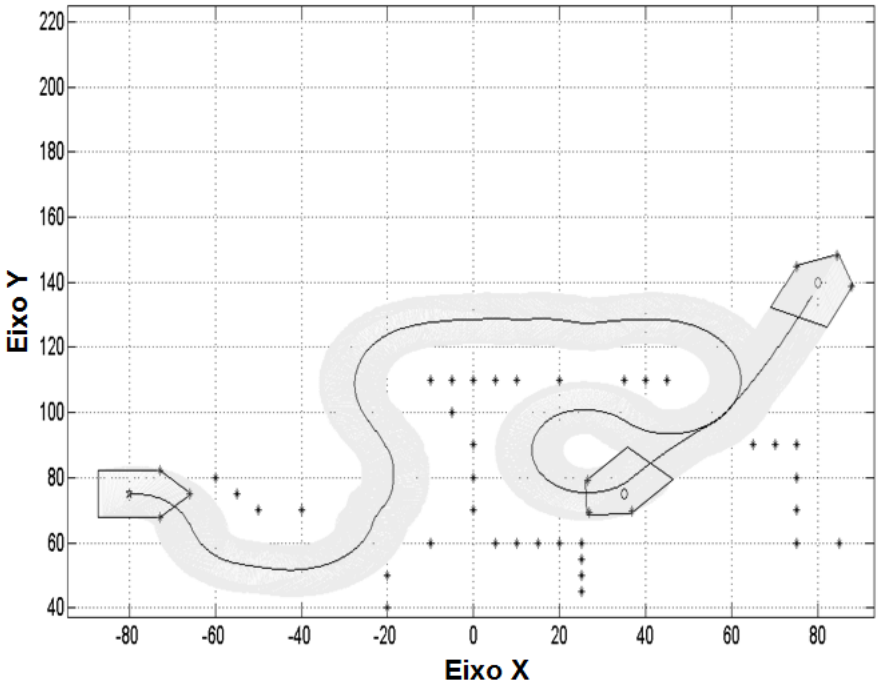


Figura 68 - Cenário 05 (d) HD-FCM.

Fonte: Autoria própria.

Por fim, foi invertida a ordem dos alvos do caso “e”, fazendo com que o robô busque primeiro a posição (80, -20) e posteriormente siga para o alvo final (35,75).

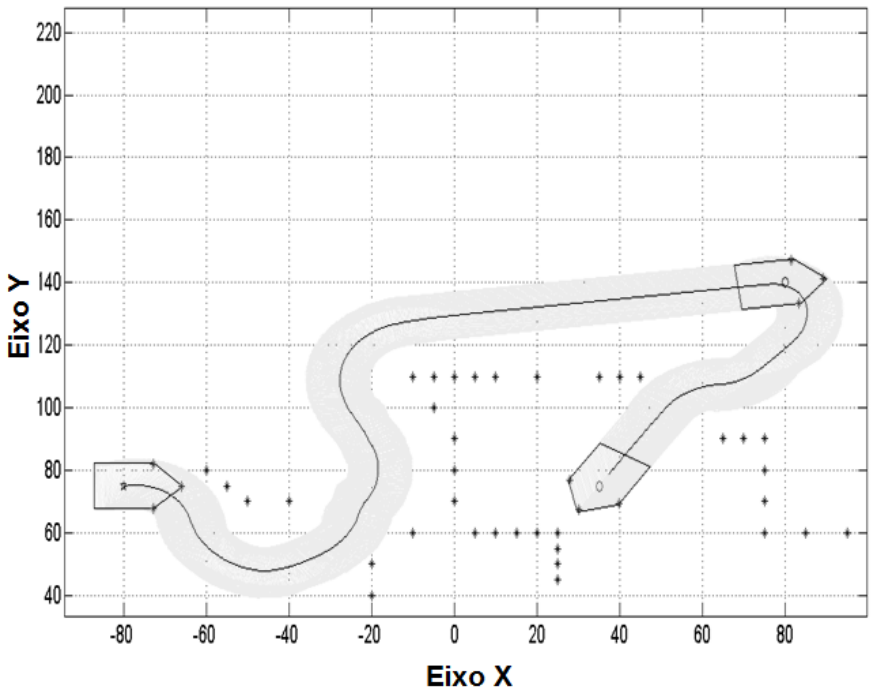


Figura 69 - Cenário 05 (e) HD-FCM.

Fonte: Autoria própria.

O robô alcançou os alvos com uma precisão de 0,15 cm depois de 874 passos. A posição ao atingir o primeiro alvo foi (80.46, 140.04), e (34.67, 74.56), a posição final.

6.3 COMPARAÇÃO ENTRE HWF E HD-FCM

Na tabela 3 serão apresentados os resultados obtidos em cada simulação comparada. Como apresentado no Capítulo 5, a WHF utiliza uma arquitetura hierárquica envolvendo Lógica *Fuzzy* e a HD-FCM também utiliza uma arquitetura hierárquica, entretanto com Mapas Cognitivos *Fuzzy*.

Tabela 3. COMPARAÇÃO HWF X HD-FCM

	Número de passos HWF	Distância posição final x alvo - HWF (cm)	Número de passos HD-FCM	Distância posição final x alvo - HD-FCM (cm)
Simulação 01-a	361	0,98	358	0,97
Simulação 01-b	473	1,95	473	1,94
Simulação 01-c	1051	1,93	1048	1,93
Simulação 02-a	549	0,82	545	0,81
Simulação 02-b	655	1,71	651	1,69
Simulação 02-c	587	1,82	584	1,82
Simulação 03-a	583	1,92	579	1,93
Simulação 03-b	397	1,43	393	1,42
Simulação 03-c	974	1,96	980	1,97
Simulação 04-a	1960	9,67	1962	9,67
Simulação 04-b	2421	9,93	2419	9,94
Simulação 05-a	936	1,68	935	1,68
Simulação 05-b	1047	5,00	1044	5,00
Simulação 05-c	1455	0,97	1458	0,95
Simulação 05-d	1022	4,84	1024	4,83
Simulação 05-e	871	0,15	874	0,15

Pode ser observado que os resultados obtidos pelas duas arquiteturas foram muito próximos com mínima vantagem para a arquitetura HD-FCM. Considerando que a velocidade é a mesma para as duas arquiteturas, a HD-FCM seria mais viável devido à facilidade de implementação, considerando a experiência do autor.

Nenhum cálculo de complexidade computacional foi realizado, a priori, essa observação é devido a menor número de operações matemáticas necessárias no FCM conforme apresentado em MENDONÇA *et al* (2015). Deste modo, esta foi a arquitetura escolhida para ser implementada.

6.4 RESULTADOS PRÁTICOS INICIAIS

Já foram implementados no Arduino os códigos iniciais para testes. O FCM destinado ao alvo obteve sucesso, e o FCM destinado a obstáculos está em fase de ajuste. Para a última atualização do software Arduino (1.6.0) a biblioteca "*Accelstepper*" utilizada na programação para ativação de dois motores de passo simultâneos não funciona, por isso, foi implementado na versão 10.5.2, penúltima versão existente em Março de 2015.

A estimativa de posição do robô foi calculada de acordo com número de passos e tamanho das rodas, devido aos motores de tração possuírem passo com boa precisão, pode-se estimar a posição atual do robô conforme procedimento apresentado no Capítulo 4.

A Figura 70 apresenta a simulação do experimento real e a Figura 71 apresenta as sequências das posições do robô em que o alvo estava localizado na posição (-80, 80), com o robô formando 180 graus em relação ao eixo "x".

Tanto no *MatLab* quanto no Arduino, foi feito uma programação inicial com lógica clássica para comparar o simulador com o robô real, e posteriormente foi implementado o FCM.

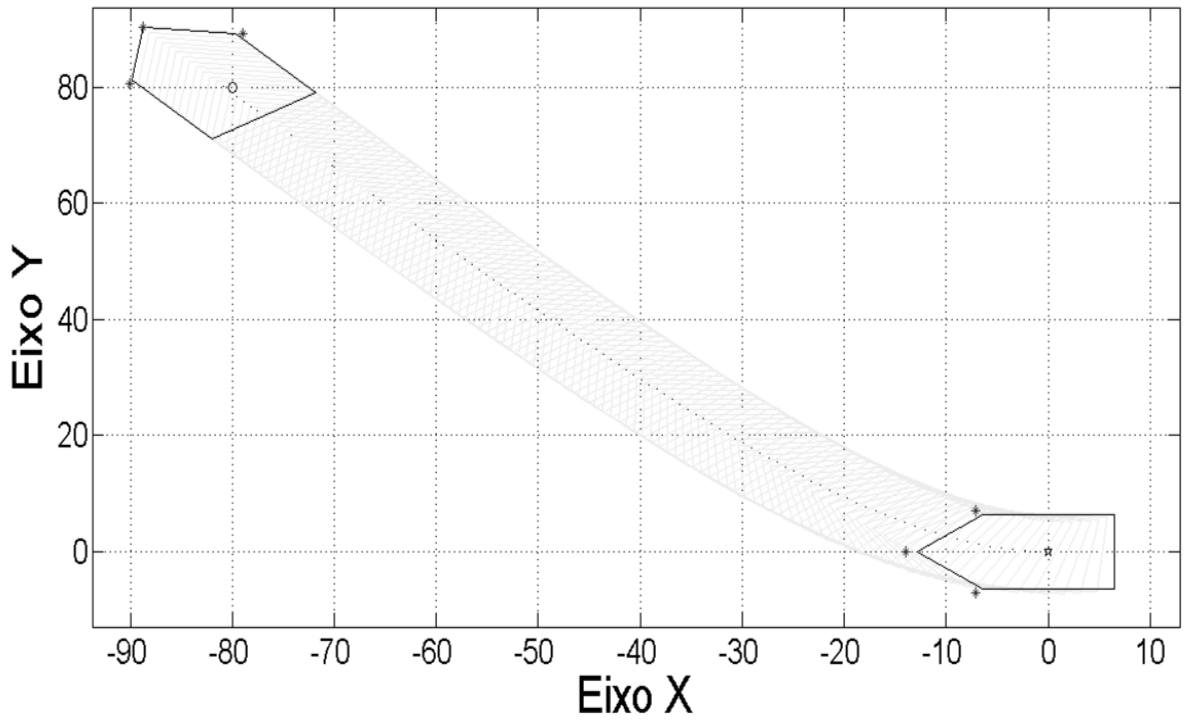


Figura 70 – Simulação - Experimento 1

Fonte: Autoria própria.

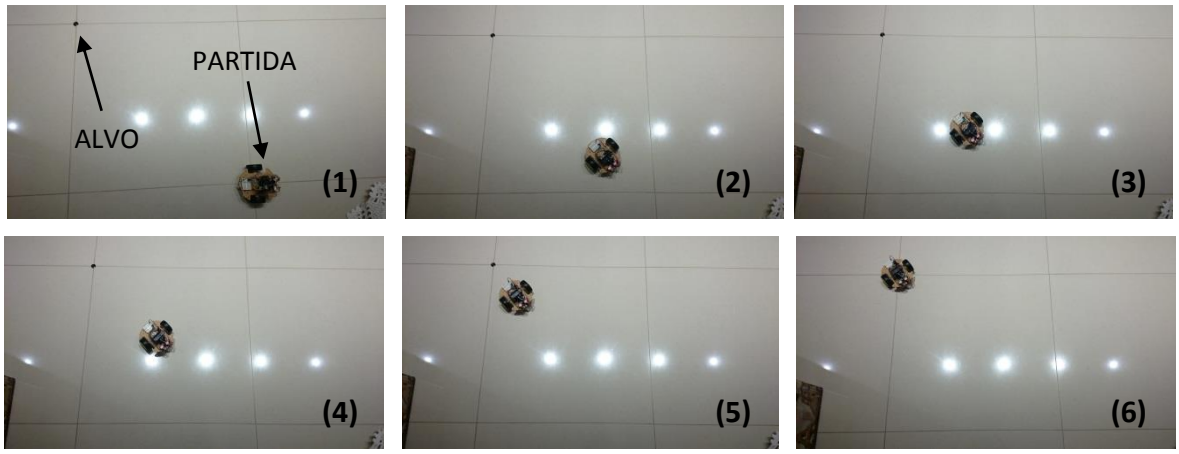


Figura 71 – Experimento Real 1

Fonte: Autoria própria.

O robô atingiu o alvo com uma precisão de 0,75 cm em um deslocamento de 113,14 cm depois de 104 passos. Ou seja, um erro de 0,66% em relação ao deslocamento.

A Figura 72 refere-se à simulação de um segundo experimento prático, no qual o alvo localiza-se na posição (-80,160) em relação ao robô, com o robô

formando 180 graus com o eixo “x”. A Figura 73 apresenta o experimento prático desta simulação.

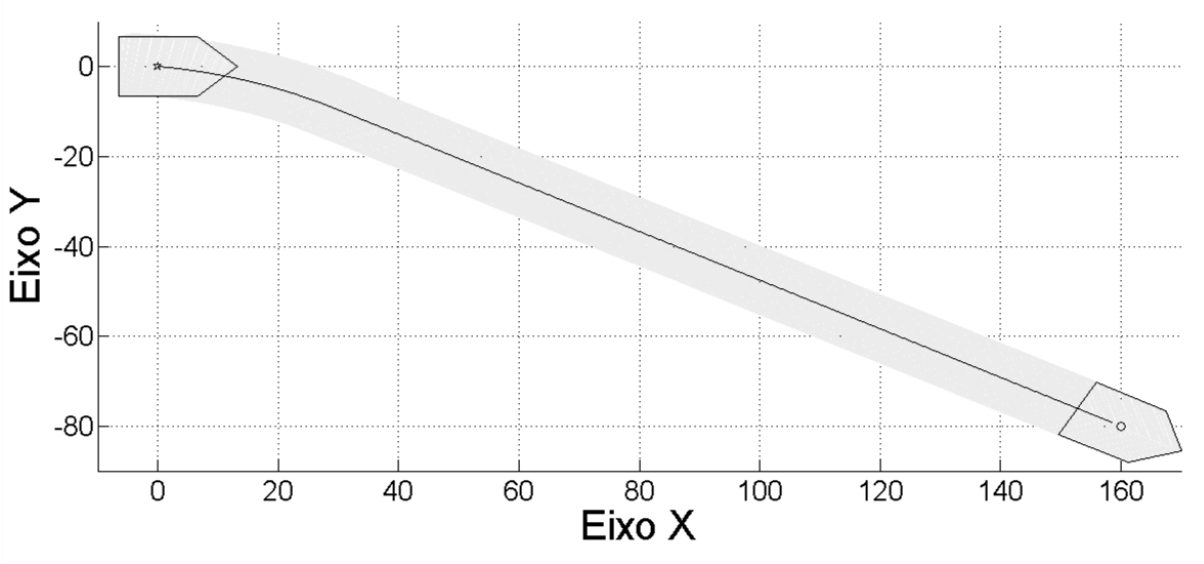


Figura 72 – Simulação Experimento 2

Fonte: Autoria própria.

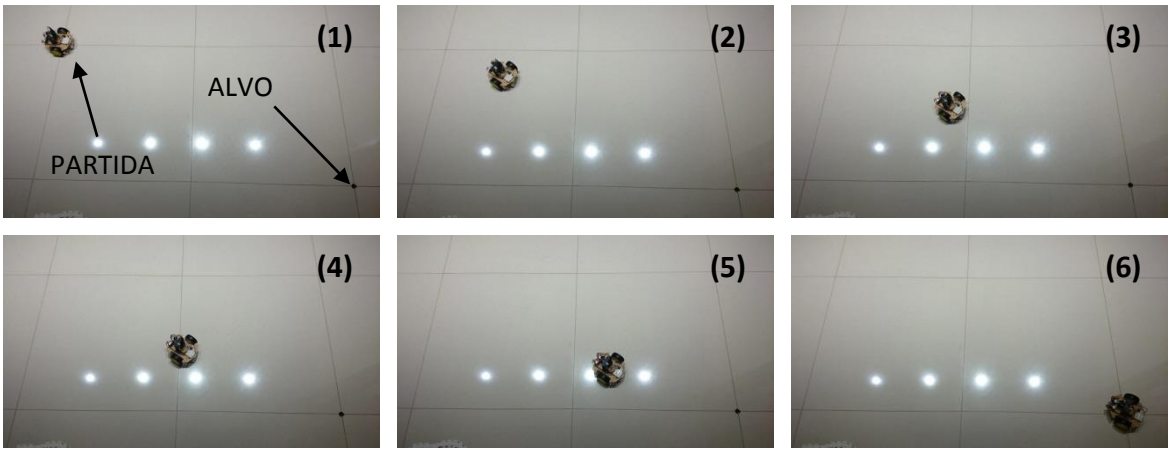


Figura 73 – Experimento Real 2

Fonte: Autoria própria.

O robô atingiu o alvo com uma precisão de 0,99 cm em um deslocamento de 178,89 cm depois de 209 passos. O que representa um erro de 0,55% em relação ao deslocamento.

Finalizando, o experimento com a trajetória discretizada da Figura 74 sugere que o modelo simulado (agente) e o robô real foram fortemente correlatos, haja vista, que a trajetória do robô foi muito próxima à trajetória simulada. Isto corroborou

com a validade do modelo cinemático utilizado nas em todas as simulações dessa dissertação, e na versão inicial do protótipo real.

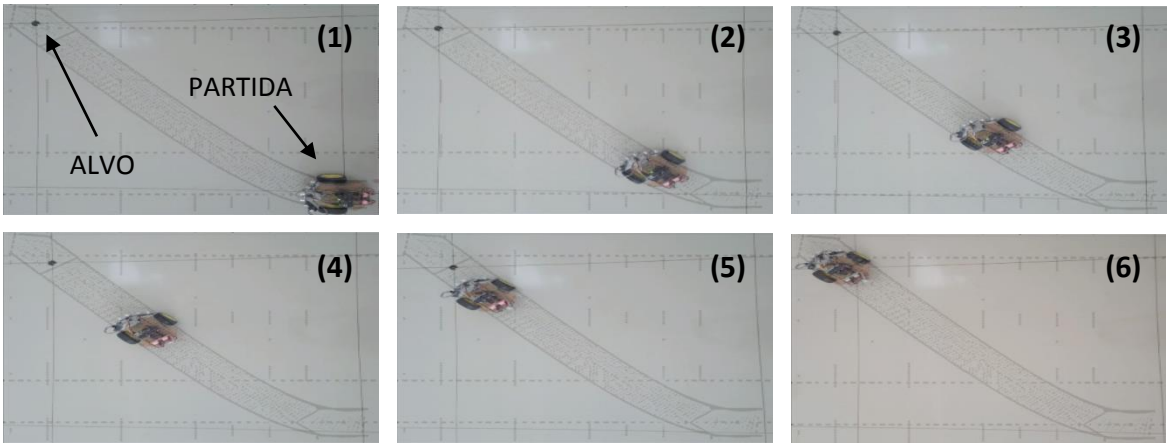


Figura 74 – Simulação e Experimento Real 3

Fonte: Autoria própria.

7 CONCLUSÕES E TRABALHOS FUTUROS

Os resultados obtidos pelas simulações foram satisfatórios, pois o agente móvel (robô) cumpriu com o objetivo de chegar ao ponto final da trajetória com o erro máximo indicado (diferença entre posição final desejada e posição alcançada) e, onde necessário, desviando de obstáculos sem nenhuma colisão nos diferentes cenários investigados.

Considerando o fato de o sistema priorizar o desvio de obstáculos, caso haja um obstáculo muito próximo do alvo, mesmo que atrás, o robô priorizaria o desvio ao invés de atingir o alvo, desta forma, em algumas situações nunca chegaria ao seu objetivo, ou faria voltas desnecessárias para isso, conforme acontece nas simulações 44c e 50c. Para casos como este, tornou-se necessário aumentar o erro desejado (distância entre posição final do robô e alvo efetivo) conforme explicado na subseção 6.1.6. Este caso ocorreu também nas simulações referentes ao cenário 4 (subseção 6.1.4 e 6.2.4). Desta forma, a diferença entre a posição final do centro de gravidade do robô e o alvo foram maiores para estes casos.

Sete ambientes diferentes foram simulados com sucesso, o que sugere autonomia dos controladores hierárquicos baseados em lógica *Fuzzy* propostos. As simulações com incertezas sugerem também robustez, pois aproximaram as simulações de cenários mais reais onde as entradas de leituras dos sensores têm a presença de ruídos e erros.

Resultados iniciais para alcance de alvos usando estimação de posição atingiram erros relativos percentualmente pequenos (menores que 01 por cento do total da trajetória). Ao menos em pequenas distâncias os resultados reais, mesmo que ainda iniciais e com apenas um objeto mais os resultados simulados sugerem que as arquiteturas de controle propostas podem ser possíveis contribuições para robótica móvel autônoma. E, finalmente, a simulação da figura 74 contribuiu de forma significativa para a validação do modelo cinemático e do protótipo dessa pesquisa.

A seguir, são apresentados trabalhos futuros para possíveis aperfeiçoamentos dessa pesquisa:

1. As comparações do controlador proposto com outras técnicas inteligentes poderão ser utilizadas para confrontar os resultados obtidos com controladores propostos, como por exemplo sistemas ANFIS;
2. Aumentar a complexidade dos cenários utilizando no sistema HWF, ajuste dinâmico dos pesos do controlador *Fuzzy*. E, nesse contexto, implementar em novas propostas de cenários com maior nível de complexidade; como por exemplo, paredes perpendiculares.
3. Embarcar o sistema com todas as funcionalidades (desvio de obstáculos e alcance de alvos) em um robô real. Através de uma plataforma de desenvolvimento *open source*, como por exemplo, um micro controlador de baixo custo (Arduino); - Já em andamento conforme apresentado na subseção 6.4;
4. Criar um sistema com número ilimitado de objetivos (alvos), incluir a função de gerenciamento de energia ou bateria, como por exemplo, para aplicações em transporte de peças.
5. Investigar a complexidade computacional de forma quantitativa dos dois controladores propostos.

REFERÊNCIAS

ACAMPORA, G.; LOIA, V. **On the Temporal Granularity in *Fuzzy* Cognitive Maps**. *Fuzzy* Systems, IEEE Transactions on, vol.19, no.6, pp.1040, 1057, Dec. 2011.

ANDERSON, J. A. **A Simple Neural Network Generating an Interactive Memory. Mathematical Biosciences**. 14:197-220. Reprinted in Anderson & Rosenfeld, 1988. pp. 181-192, 1972.

ARRIAGA-DE-VALLE, E.; DIECK-ASSAD, G. **Modeling and Simulation of a *Fuzzy* Supervisory Controller for an Industrial Boiler.**2006. Disponível em: < http://sim.sagepub.com/content/82/12/841.short>. Acesso em: 03 nov. 2014

AXELROD, R. **Structure of decision: the cognitive maps of political elites**. New Jersey: Princeton University Press, 1976.

BAKAMBU, J. N. **Integrated autonomous system for exploration and navigation in underground mines.** Proc. 2006 IEEE/RSJ Int. Conf. Intelligent Robots and Systems, pp.2308 -2313, 2006.

BLOCH,A. M.; REYHANOGLU, M.; MCCLAMROCH N. H. **Control and stabilization of nonholonomic dynamic sytems**. IEEE Transactions on Automatic Control, 37(11):1746–1756, Nov 1992.

BOSE, N. K.; LIANG, P. **Neural network fundamentals with graphs, algorithms, and applications**. New York: McGraw-Hill, 1996.

BRAITENBERG, V. **Vehicles: Experiments in Synthetic Psychology**. The mit Press, 1986.

BROGGI, A.; ZELINSKY, A.; PARENT, M.; THORPE, C. E. **Intelligent vehicles**. Springer Handbook of Robotics, 1175–1198, 2008.

BROOKS, R. **A robust layered control system for a mobile robot**. Robotics and Automation, IEEE Journal of 2(1), 14 – 23, 1986.

BUCHE C.; CHEVAILLIER, P.; NÉDÉLEC, A.; PARENTHOËN M. & TISSEAU J. ***Fuzzy*** **cognitive maps for the simulation of individual adaptive behaviors**. Computer Animation and Virtual Worlds, Volume 21, pp 573–587, 2010

CAI, L.; RAD, A.; CHAN, W. L. **An intelligent longitudinal controller for application semiautonomous vehicles**. IEEE Transactions on Industrial Electronics 57(4), 1487–1497, 2010.

CALVO, R. **Arquitetura híbrida inteligente para navegação autônomo de robôs**. Dissertação (Mestrado em Ciências de Computação e Matemática Computacional). IMC-USP, 2007.

_____; ROMERO, R. A. F. **A Hierarchical self-organizing controller for navigation of mobile robots**. Proceedings of International Joint Conference on Neural Netwoks. Vancouver: Ieee World Congress Computational Intelligence, 2006.

CALLAI, T. C.; COELHO L. S.; COELHO A. A. R.. **Controle nebuloso adaptativo por modelo de referência: projeto e aplicação em sistemas não lineares**. SBA Controle & Automação, Campinas, v.18, n. 4, p. 479-489, Oct./Dec. 2007.

CHANGEUX, J.P; DANCHIN, A.. **Selective stabilization of developing synapses as a mechanism for the specification of neural networks**, Nature, vol. 264. pp. 705-712, 1976.

CHIANG, H.-H.; MA, L.-S., PERNG, J. W.; WU, B. F. **Longitudinal and lateral** ***Fuzzy*** **control systems design for intelligent vehicles**. Proceedings of the 2006 IEEE InternationalConference on Networking, Sensing and Control (ICNSC), pp. 544–549, 2006.

CHUNMEI, Lin. **Using** ***Fuzzy*** **Cognitive Map for System Control**, Wseas Transactions On Systems, pp.12, 2008.

COPPIN, B. **Inteligência Artificial**. Rio de Janeiro: LTC, 2010.

COSTA, E.D.S.; GOUVEA, M.M. **Autonomous Navigation in Dynamic Environments with Reinforcement Learning and Heuristic**. Machine Learning and Applications (ICMLA), 2010 Ninth International Conference on, vol., no., pp.37, 42, 12-14 Dec. 2010.

CRAIG, J. J. **Introduction to Robotics: Mechanics and Control (2nd Edition)**. Addison-Wesley, 1989.

CUNHA, A. G.; TAKAHASHI, R.; ANTUNES, C. H. **Manual de Computação Evolutiva e Metaheurística**. Belo Horizonte, Editora UFMG, Coimbra, 2013.

DA SILVA, A. M. **Sistemas Neuro-*Fuzzy* Evolutivos: Novos Algoritmos de Aprendizado e Aplicações**. Tese de doutorado, Universidade Federal de Minas Gerais - Programa de Pós-Graduação em Engenharia Elétrica, Belo Horizonte, 2014.

DICKERSON, J. A., KOSKO, B. **Virtual worlds as *Fuzzy* cognitive maps.** Presence, v. 3, n. 2, p. 173-189, 1994.

_____; KOSKO, B. **Virtual worlds as *Fuzzy* dynamical systems. In:** Sheu, B. (Ed.) Technology for multimedia, New York: IEEE Press, 1996.

DOERSCHUK, P.; JIANGJIANG Liu; MANN, J. **An INSPIRED game programming academy for high school students.** Frontiers in Education Conference (FIE), On pp. 1 – 6, 2012.

FARHI, O.; CHERVENKOV, Y. **Optimal *Fuzzy* control of autonomous robot car**. Intelligent Systems, 2008. IS '08. 4th International IEEE Conference, pp. 4.62 – 4.65. 2008.

FIGUEIREDO, L. C; ALMEIDA; P.E.M., BRAGA A.R.; JOTA, F.G.; ARAÚJO, E.O. **Ambiente integrado para análise e desenvolvimento de controladores difusos**. In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 1. p. 291-299. Rio Claro, 1993.

FIGUEIREDO, L. C. & TEIXEIRA, R.A., **"Implementação de um Controlador *Fuzzy* em CLP"**, II Congresso Mineiro de Automação, V Simpósio Regional de Instrumentação, Belo Horizonte - MG, pp. 73-79, Agosto, 1998.

_____; ALMEIDA; P.E.M., BRAGA A.R.; JOTA, F.G.; ARAÚJO, E.O**. Ambiente integrado para análise e desenvolvimento de controladores difusos.** In: SIMPÓSIO BRASILEIRO DE AUTOMAÇÃO INTELIGENTE, 1. p. 291-299. Rio Claro, 1993.

FRACASSO, P. T.; COSTA, A. H. R. **Navigation From Reactive Autonomous Mobile Robots Using *Fuzzy* Logic Rules With Weighted**. LTI – Escola Politécnica da Universidade de São Paulo, Laboratório de Técnicas Inteligentes, 2005.

FRANZ A. SANDI L.; ELDER M. H. & WALTER F. L. **Sistema Para Navegação e Guiagem De Robôs Móveis Autônomos**. SBA Controle & Automação Vol. 9 no. 3, Pp 107 – 118, Set., Out., Nov. e Dezembro de 1998.

GANGANATH, N. ; WALKER, M. ; LEUNG, H. ***Fuzzy* Cognitive Map Based Situation Assessment Framework for Navigation Goal Detection**. Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference (2013)

GARNIER, P.; NOVALES, C.; Laugier, C. **An hybrid motion controller for a real car-like robot evolving in a multi-vehicle environment**, Proceedings of the Intelligent Vehicles' 95 Syposium, pp. 326 – 331, Detroit, USA.

GLYKAS, Michael. **Fuzzy Cognitive Maps** Advances in Theory, Methodologies, Tools and Applications. Greece Springer: 2010.

GOERICK, C. **Towards An Understanding Of Hierarchical Architectures**. IEEE Trans. Auton. Ment. Dev. 3, 54–63, 2011.

GOLMOHAMMADI, S.K.; AZADEH, A.; GHAREHGOZLI, A. **Action Selection in Robots Based on Learning *Fuzzy* Cognitive Map.** IEEE Conferência Internacional sobre Informática Industrial, 2006

GOMIDE, F.; PEDRYCZ, W. ***Fuzzy*** **Systems Engineering Toward Human-Centric Computing**. IEEE Press, 2007.

HAYKIN, S. **Neural Networks: a comprehensive foundation**. 842p. Prentice-Hall, Inc, 2a. ed. New Jersey, 1999.

_____. **Redes neurais, princípios e prática, 2. ed**. São Paulo: Bookman, 2000.

HEBB, D. O. **The Organization of Behavior**. John Wiley & Sons, New York. Introduction and Charpter 4, 1949. Reprinted in Anderson & Rosenfeld, pp. 45 – 56, 1988.

HENKEL, Z.; DOERSCHUK, P.; MANN, J. **Exploring Computer Science through Autonomous Robotics**, 39th ASEE/IEEE Frontiers in Education Conference, San Antonio, TX, pp. 18-21, October 2009.

KHODAYARI, A. G. **A historical review on lateral and longitudinal control of autonomous vehicle motions. Singapore**: Mechanical and Electrical Technology (ICMET), 2nd International Conference on, 2010.

KING, P. J.; MAMDANI, E. H., **The application of Fuzzy control systems to industrial process.** Automática, v. 13, n. 3, p. 235-242, May 1977.

KOHONEN, T. **Correlation Matrix Memories**. IEEE Transaction on Computers, C-21:353-359, 1987.

KOLMANOVSKY, I.; MCCLAMROCH N. H. **Developments in nonholonomic control problems**. IEEE Control Systems Magazine, 15(6):20–36, 1995.

KONOLIGE, K.; MYERS, K. **The Saphira Architecture for Autonomous Mobile Robots**. In: Artificial intelligence and mobile robotics: case studies of successful robot systems. Kortenkamp, D.; Bonasso, R.P.; Murphy, R. P. 211 – 242, Cambridge: MIT Press, 1998.

KOSKO, B. **Fuzzy cognitive maps. International Journal Man-Machine Studies**, v. 24, n. 1, p.65-75, 1986.

KOSKO, B. **Neural networks and fuzzy systems: a dynamical systems approach to machine intelligence**. New York: Prentice Hall, 1992.

LEE, K. C.;LEE, S. **A cognitive map simulation approach to adjusting the design factors of the electronic commerce web sites**. Expert Systems with Applications, v. 24, n. 1, p. 1-11, Jan. 2003.

LLORCA, D.F. ; MILANES, V. ; ALONSO, I.P. ; GAVILAN, M.; DAZA, I. G., PÉREZ, J.; SOTELO, M. A., **Autonomous Pedestrian Collision Avoidance Using a *Fuzzy* Steering Controller**. IEEE Transactions on Intelligent Transportation Systems. pp. 390 – 401. (2011).

LYONS, d. M.; HENDRIKS, A. **Planning as incremental adaptation of a reactive system**. Robotics and Autonomous Systems, v. 14, n. 4, p. 255–288, 1995.

MAMDANI, E. H. **Application of *Fuzzy* algorithms for the control of a simple dynamic plant.** In Proc IEEE, 121-159, 1974.

_____; Assilian, S. "**An Experiment in Linguistic Synthesis with a *Fuzzy* Logic Controller**", Int. J. Man-Machine Studies, 7, pp. 1-13, 1974.

MATARIĆ, J.M. **The Robotics Primer**. The MIT Press, 2007.

MCCLELLAND, J. L.; RUMELHART, D. E. **Exploration in Parallel Distributed Processing**. MA: MIT Press Cambridge, 1988.

MENDONÇA, M. **Uma Contribuição ao Desenvolvimento de Sistemas Inteligentes Utilizando Redes Cognitivas Dinâmicas**. Universidade Tecnológica Federal do Paraná, campus Curitiba, 2011.

_____; ANGÉLICO, B. M.; ARRUDA, L. V. R.; NEVES, F. A.. **A Subsumption Architecture to Develop Dynamic Cognitive Network-Based Models with**

Autonomous Navigation Application. Journal of Control, Automation and Electrical Systems, vol. 1, pp. 3-14. 2013.

_____; ARRUDA, L. V. R.; NEVES, F. A. **Autonomous Navigation System Using Event Driven-*Fuzzy* Cognitive Maps.** Applied Intelligence (Boston), vol. 37, pp. 175-188, 2011.

_____; _____; CHRUN, I. R.; PAPAGEORGIOU, E. **Autonomous Navigation Applying Dynamic-*Fuzzy* Cognitive Maps and *Fuzzy* Logic**. In: 9th IFIP International Conference on Artificial Intelligence Applications and Innovations (AIAI 2013), 2013, Chipre. 9th IFIP International Conference on Artificial Intelligence Applications and Innovations (AIAI 2013), 2013.

_____; _____; _____; SILVA, E. S. **HYBRID AUTONOMOUS NAVIGATION SYSTEM USING A DYNAMIC FUZZY COGNITIVE MAPS EVOLUTION**. ICAS – International Conference on Autonomic and Autonomous Systems, pp 114-119, 2015

MIAO, Y., LIU Z-Q, SIEW, C. K., and MIAO C. Y. **Dynamical cognitive network: an extension of Fuzzy cognitive map**. IEEE Transactions on Fuzzy Systems, v. 9, n. 5, p. 760-770, 2001.

MIAO, Y. MIAO C. Y. TAO X., SHEN Z. LIU Z. **Transformation of cognitive maps**. IEEE Transactions on Fuzzy Systems, v. 18, n. 1, p. 114-124, Feb. 2010.

MOTLAGH O.; TANG, S.H.; ISMAIL, N.; RAMLI, A.R. **An expert *Fuzzy* cognitive map for reactive navigation of mobile robots**. *Fuzzy* Sets and Systems, Volume 201, pp 105–121, 2012.

MURATA S.; HIROSE T. **On Board Locating System Using Real–Time Image Processing for a Self–Navigating Vehicle**. IEEE Transactions on Industrial Electronics, 40(1):145–153, 1993.

NOGUEIRA, M. M. **Aplicando lógica *Fuzzy* no controle de robôs móveis usando dispositivos lógicos programáveis e a linguagem VHDL.**. Dissertação (mestrado)

- Universidade Estadual Paulista Júlio de Mesquita Filho, Faculdade de Engenharia de Ilha Solteira, 95 f., 2013. Disponível em: <http://hdl.handle.net/11449/87069>.

PADILHA, P. C. C. **Desenvolvimento de uma metodologia de sintonia de controladores "*Fuzzy*" utilizando redes neurais**. Aplicações em processos petroquímicos, 2001, Dissertação Mestrado. Instituto Militar de Engenharia**,** Rio de Janeiro, 2001.

PAJARES, G.; DE LA CRUZ, J. M. ***Fuzzy* cognitive maps for stereovision matching** Pattern Recognition, v. 39, n. 11, p. 2101-2114, Nov. 2006.

PAPAGEORGIOU, E. I. ***Fuzzy* Cognitive Maps for Applied Sciences and Engeneering**: From Fundamentals to Extensions and learning Algorithms. Intelligent Systems Reference Library 54. Springer, Heidelberg, 2014.

_____. **Learning Algorithms for *Fuzzy* Cognitive Maps**. IEEE Transactions ON Systems and Cybernetics. Part C: Applications and Reviews. Vol 42 pag. 150-163, Mar. 2012.

_____; STYLIOS, C.; GROUMPOS, P. **Novel for supporting medical decision making of different data types based on *Fuzzy* Cognitive Map Framework**. Proceedings of The 29th Annual International Conference Of The IEEE Embs Cité Internationale, Lyon, France August 23-26, 2007

PASSINO, M. K.; YOURKOVICH S. ***Fuzzy* control**. Menlo Park: Addison-Wesley. 1997. Impresso, 1998.

PERUSICH, K. ***Fuzzy* cognitive maps for policy analysis**. Ieee Purdue University South Bend, IN 46634 USA.1996.

POLICASTRO, C.A.; ROMERO, R.A.F.; ZULIANI, G. **Robotic Architecture Inspired on Behavior Analysis**, Neural Networks, 2007. IJCNN 2007. International Joint Conference on, vol., no., pp.1482, 1487, 12-17 Aug. 2007.

RAMIREZ, J. M.; GOMEZ-GIL, P.; LOPEZ LARIOS, F. **A Robot-vision System for Autonomous Vehicle Navigation with *Fuzzy*-logic Control using Lab-View. Electronics.** Robotics and Automotive Mechanics Conference. CERMA, pp 295-302. 2007

RASSHOFER, R. H.; GRESSER K. **Automotive Radar and Lidar Systems for Next Generation Driver Assistance Functions**. Adv. Radio Sci., 3, 205-209, doi:10.5194/ars-3-205-2005, 2005.

ROCHESTER, N.HOLLAND, J. H.; HAIBT L, H.; DUDA, W. L. **Test on a Cell Assembly Theori of the Action of the Brain, Using a Large Digital Computer**. IRE Transactions on Information Theory. IT-2:80-93. Reprinted in Anderson & Rosenfeld, 1988. pp. 68-80, 1956.

ROMERO, R. A. F. et (orgs.). **Robótica Móvel**. LTC, 07/2014. Vital Book, 2014

RUSSELL, S. J.; NORVIG, P. **Artificial Intelligence**: A Modern Approach. Englewood Cliffs: Prentice Hall, 1995.

SAFFIOTTI, A. **The uses of *Fuzzy* logic in autonomous robot navigation**: a catalogue Raisonné. Soft Computing Research Journal, vol. 1, p. 180 – 197, 1997.

SANDI L.; F. A.; HEMERLY, E. M.; LAGES, W. F. **Sistema para navegação e guiagem de robôs móveis autônomos**. Revista SBA Controle e Automação, Campinas, v. 9, n. 3, p. 107-118, 1998. Disponível em: <>. Acesso em: 12 abr. 2013.

SHAIKH, M.H.; KOSURI, K.; ANSARI, N.A.; KHAN, M.J. **The state-of-the-art intelligent navigational system for monitoring in mobile autonomous robot**. Information and Communication Technology (ICoICT), 2013 International Conference of, vol., no., pp.405,409, 20-22, 2013.

SILVA, I. N.; SPATTI, D. H.; FLAUZINO, R. A. **Redes neurais artificiais: para engenharia e ciências aplicadas - curso prático**. São Paulo: Artiliber, 2010.

STENT, G.S. **A physiological mechanism for Hebb´s postulate of learning**, Proceeding o th National Academy of Sciences, USA, vol 70, pp. 997-1001, 1973.

STYLIOS, C. D., GEORGEOPULOS, V.C. MALANDRAKI, G.A., CHOULIARA S. ***Fuzzy*** **cognitive map architectures for medical decision support systems. Applied Soft Computing**, v. 8, n. 3, p. 1243-1251, Jun. 2008.

_____; _____; GROUMPOS, P. P. **Introducing the theory of *Fuzzy* cognitive maps in distributed systems**. PROCEEDINGS OF THE 12TH IEEE INTERNATIONAL SYMPOSIUM ON INTELLIGENT CONTROL, Istanbul, Turkey, 1997, pp. 55–60.

_____; GEORGOPOULOS, V. C.; GROUMPOS, P. P. **The use of *Fuzzy* cognitive maps in modeling systems**. In: 5th IEEE MEDITERRANEAN CONFERENCE ON CONTROL AND SYSTEMS, PAPHOS, Cyprus, 21-23 July 1997.

_____; GROUMPOS, P. P. A. **Soft computing approach for modeling the supervisor of manufacturing systems**. Journal of Intelligent and Robotic Systems, v. 26, n. 3-4, p. 389-403, Nov. 1999b.

_____; _____. ***Fuzzy*** **cognitive maps in modeling supervisory control systems. Journal of Intelligent &** ***Fuzzy*** **Systems**, v. 8, n. 2, p. 83–98, 2000.

_____; _____. ***Fuzzy*** **cognitive maps: a model for intelligent supervisory control systems**. Computers in Industry, v. 39, n. 3, p. 229-238, 1999.

_____; _____. **The challenge of modeling supervisory systems using *Fuzzy* cognitive maps**. Journal of Inteligent Manufaturing, 9, 339-345, 1998.

_____; GROUMPOS, P. P: **Modeling complex systems using *Fuzzy* cognitive maps**. IEEE Transactions on Systems, Man, and Cybernetics, Part A 34(1): 155-162, 2004

TABER, R. ***Fuzzy*** **cognitive maps model social systems**. AI Expert, v. 9, p. 18-23, 1994.

WANG, L. **Analysis and design of hierarchical *Fuzzy* systems**. IEEE Trans. *Fuzzy* Syst., vol. 7, pp. 617–624, 1999

WOOLDRIDGE, M.J.; JENNINGS, N. R. **Intelligent Agents:** Theory and Practice Knowledge Engineering Review Vol. 10 No. 2 Cambridge University Press pp. 115-152, 1995.

YANG, J.; ZHENG, N. **An expert *Fuzzy* controller for vehicle lateral control**. 33rd Annual Conference of the IEEE on Industrial Electronics Society (IECON), pp. 880–885, 2007.

ZADEH, L. A. ***Fuzzy* algorithms, Info. & Ctl**. Vol. 12, pp. 94-102, 1968.

_____. ***Fuzzy* sets**. Information and Control. 330-353, pp.1965.

APÊNDICES

APÊNDICE A – Tabela 1 completa

Tabela 1. Regras do *Fuzzy* I

1. If (DSx is farE) and (DSy is farF) then (we is medium)(wd is high) (0.5)
2. If (DSx is farE) and (DSy is mediumF) then (we is slow)(wd is high) (0.5)
3. If (DSx is farE) and (DSy is closeF) then (wd is high) (0.5)
4. If (DSx is mediumE) and (DSy is farF) then (we is medium)(wd is high) (0.5)
5. If (DSx is mediumE) and (DSy is mediumF) then (we is slow)(wd is high) (0.5)
6. If (DSx is mediumE) and (DSy is closeF) then (wd is high) (0.5)
7. If (DSx is closeE) and (DSy is farF) then (we is medium)(wd is high) (0.5)
8. If (DSx is closeE) and (DSy is mediumF) then (we is slow)(wd is medium) (0.5)
9. If (DSx is closeE) and (DSy is closeF) then (wd is slow) (0.5)
10. If (DSx is farD) and (DSy is farF) then (we is high)(wd is medium) (0.5)
11. If (DSx is farD) and (DSy is mediumF) then (we is high)(wd is slow) (0.5)
12. If (DSx is farD) and (DSy is closeF) then (we is high) (0.5)
13. If (DSx is mediumD) and (DSy is farF) then (we is high)(wd is medium) (0.5)
14. If (DSx is mediumD) and (DSy is mediumF) then (we is high)(wd is slow) (0.5)
15. If (DSx is mediumD) and (DSy is closeF) then (we is high) (0.5)
16. If (DSx is closeD) and (DSy is farF) then (we is high)(wd is medium) (0.5)
17. If (DSx is closeD) and (DSy is mediumF) then (we is medium)(wd is slow) (0.5)
18. If (DSx is closeD) and (DSy is closeF) then (we is slow) (0.5)
19. If (DSy is closeB) then (we is high) (0.5)
20. If (DSy is mediumB) then (we is high)(wd is slow) (0.5)

21. If (DSy is farB) then (we is high) (wd is medium) (0.5)
22. If (DSx is farE) then (we is medium)(wd is high) (0.5)
23. If (DSx is mediuE) then (we is slow)(wd is high) (0.5)
24. If (DSx is closeE) then (wd is high) (0.5)
25. If (DSx is closeD) then (we is high) (0.5)
26. If (DSx is mediumD) then (we is high)(wd is slow) (0.5)
27. If (DSx is farD) then (we is high)(wd is medium) (0.5)
28. If (DSy is closeF) then (we is slow)(wd is slow) (0.5)
29. If (DSy is mediumF) then (we is medium)(wd is medium) (0.5)
30. If (DSy is farF) then (wd is high) (we is high) (0.5)
31. If (DSx is farE) and (DSy is farB) then (we is medium)(wd is high) (0.5)
32. If (DSx is farE) and (DSy is mediumB) then (we is slow)(wd is high) (0.5)
33. If (DSx is farE) and (DSy is closeB) then (wd is high) (0.5)
34. If (DSx is mediumE) and (DSy is farB) then (we is medium)(wd is high) (0.5)
35. If (DSx is mediumE) and (DSy is mediumB) then (we is slow)(wd is high) (0.5)
36. If (DSx is mediumE) and (DSy is closeB) then (wd is high) (0.5)
37. If (DSx is closeE) and (DSy is farB) then (we is medium)(wd is high) (0.5)
38. If (DSx is closeE) and (DSy is mediumB) then (we is slow)(wd is medium) (0.5)
39. If (DSx is closeE) and (DSy is closeB) then (wd is slow) (0.5)
40. If (DSx is farD) and (DSy is farB) then (we is high)(wd is medium) (0.5)
41. If (DSx is farD) and (DSy is mediumB) then (we is high)(wd is slow) (0.5)
42. If (DSx is farD) and (DSy is closeB) then (wd is high) (0.5)
43. If (DSx is mediumD) and (DSy is farB) then (we is high)(wd is medium) (0.5)
44. If (DSx is mediumD) and (DSy is mediumB) then (we is high)(wd is slow) (0.5)
45. If (DSx is mediumD) and (DSy is closeB) then (wd is high) (0.5)
46. If (DSx is closeD) and (DSy is farB) then (we is high)(wd is medium)

(0.5)
47. If (DSx is closeD) and (DSy is mediumB) then (we is medium)(wd is slow) (0.5)
48. If (DSx is closeD) and (DSy is closeB) then (we is slow) (0.5)

APÊNDICE B – Tabela 2 completa.

Tabela 2. Exemplo de regras do *Fuzzy* II

1. If (sensor_E is far) and (sensor_F is far) then (we is high) (1)
2. If (sensor_E is far) and (sensor_F is medium) then (we is high) (1)
3. If (sensor_E is far) and (sensor_F is close) then (we is high) (1)
4. If (sensor_E is far) and (sensor_F is far) then (we is high) (1)
5. If (sensor_E is far) and (sensor_F is medium) then (we is high) (1)
6. If (sensor_E is far) and (sensor_F is close) then (we is high) (1)
7. If (sensor_E is far) and (sensor_F is far) then (we is high) (1)
8. If (sensor_E is far) and (sensor_F is medium) then (we is high) (1)
9. If (sensor_E is far) and (sensor_F is close) then (we is high) (1)
10. If (sensor_F is far) and (sensor_D is far) then (wd is high) (1)
11. If (sensor_F is far) and (sensor_D is medium) then (wd is high) (1)
12. If (sensor_F is far) and (sensor_D is close) then (wd is high) (1)
13. If (sensor_F is far) and (sensor_D is far) then (wd is high) (1)
14. If (sensor_F is far) and (sensor_D is medium) then (wd is high) (1)
15. If (sensor_F is far) and (sensor_D is close) then (wd is high) (1)
16. If (sensor_F is far) and (sensor_D is far) then (wd is high) (1)
17. If (sensor_F is far) and (sensor_D is medium) then (wd is high) (1)
18. If (sensor_F is far) and (sensor_D is close) then (wd is high) (1)
19. If (sensor_E is far) then (we is low) (1)
20. If (sensor_E is far) then (we is medium) (1)
21. If (sensor_E is far) then (we is high) (1)
22. If (sensor_D is far) then (wd is low) (1)
23. If (sensor_D is far) then (wd is medium) (1)
24. If (sensor_D is far) then (wd is high) (1)