UNIVERSIDADE FEDERAL DA BAHIA

INSTITUTO DE MATEMÁTICA

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

JESSICA MEYER GARCIA

CONTROLADORES FUZZY PARA O POSICIONAMENTO DO GOLEIRO EM RELAÇÃO AO ATACANTE NO FUTEBOL DE ROBÔS SIMULADO 2D

Salvador

2007

JESSICA MEYER GARCIA

CONTROLADORES FUZZY PARA O POSICIONAMENTO DO GOLEIRO EM RELAÇÃO AO ATACANTE NO FUTEBOL DE ROBÔS SIMULADO 2D

Monografia apresentada ao Curso de Bacharelado em Ciência da Computação, Departamento de Ciência da Computação, Instituto de Matemática, Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Marco Antônio Costa Simões
Co-orientador: Augusto da Costa Loureiro

Salvador
2007

AGRADECIMENTOS

Agradeço primeiramente a minha mãe, Lúcia Luz Meyer, por estar sempre ao e do meu lado durante toda minha vida. Aos meus avós Affonso e Walda, pela presença fundamental em minha infância e pelo exemplo de retidão de caráter. Com minha família adquiri o gosto pela ficção científica, desde berço assistindo e ouvindo histórias extraordinárias, o que com certeza influenciou em minha escolha pela área da Robótica Inteligente.

Agradeço ao meu tio Carlos pelo apoio e torcida, e aos meus primos Ygon, Egon e Eric. À minha irmã Marina pela amizade especial. Também aos demais irmãos, primos, tios e ao meu pai.

Não poderia deixar de agradecer aos grandes mestres que me acompanharam durante esta jornada. Ao meu orientador, Marco Simões, pelo apoio e incentivo, e por ter criado a oportunidade para que eu fizesse Iniciação Científica, publicasse artigos, participasse de competições - tanto a nível nacional como internacional. Ao meu co-orientador Augusto Loureiro pelo apoio e compreensão. À professora Frieda pelo apoio e disponibilidade junto ao Colegiado; assim como a Kléber, pela presteza e cordialidade.

Agradeço também aos meus colegas de Iniciação Científica, aos meus colegas de semestre e das turmas de 2002.1 e 2002.2.

E, por fim, aos meus verdadeiros amigos ainda não citados aqui e que contribuíram para minha formação.

It is the ability of humans to think and reason in imprecise, non quantitative terms that makes it possible for humans to decipher sloppy hand-writing, understand distorted speech, and focus on that information that is relevant to a decision. It is the lack of this ability that makes even the most sophisticated computer incapable of communicating with humans in natural - rather than artificially constructed - languages.

Lotfi Zadeh, inventor da Teoria de Conjuntos Fuzzy.

RESUMO

Este trabalho descreve uma pesquisa sobre controladores fuzzy para o posicionamento do goleiro em relação ao atacante no futebol de robôs simulado em duas dimensões. Tal pesquisa foi realizada com o intuito de melhorar o desempenho do goleiro, mais especificamente, aumentar a quantidade de bolas agarradas pelo mesmo, em detrimento aos gols sofridos, melhorando seu posicionamento para recepcionar a bola. Como o futebol trata de um ambiente dinâmico e impreciso, a lógica fuzzy adequou-se bem ao problema. Construiu-se um controlador para o posicionamento na coordenada Y do goleiro e, posteriormente, outro para a coordenada X. Os controladores sofreram vários ajustes ao longo do trabalho, cada vez ficando mais eficientes, até gerar uma versão final. Eles foram implementados utilizando-se a ferramenta gráfica Xfuzzy. Para validar a pesquisa, foram simuladas 20 partidas contra alguns dos melhores times do mundo que participam da RoboCup. Os resultados obtidos são apresentados e as conclusões tomadas são discutidas, bem como são oferecidas sugestões de trabalhos futuros.

Palavras-chave: Lógica Fuzzy; Futebol de Robôs; Agentes Inteligentes; Robôs Autônomos; RoboCup.

ABSTRACT

This work describes a research about fuzzy controllers for the positioning of the goalkeeper in relation to the attackers in the 2d simulated robot soccer. The goal of this research was to improve the behavior of the goalkeeper, raising the number of catches and decreasing the number of goals, improving its positioning to catch the ball. As the game soccer deals with a dynamic and imprecise environment, fuzzy logic adjusted well to the problem. A fuzzy controller was originally built for the Y coordinate of the goalkeeper, and than another one for the X coordinate. The controllers suffered severals adjustments during the research, getting more and more efficients, until it generated a final version. They were implemented with the help of the graphical tool Xfuzzy. To validate the research, 20 matches were simulated against some of the best teams of the world that participate of the RoboCup. The achieved results are presented, the conclusions are discussed and future works are suggested.

Key Words: Fuzzy Logic; Robot Soccer; Intelligent Agents; Autonomous Robots; RoboCup.

LISTA DE FIGURAS

Figura 1. Esquema de um Agente. 16

Figura 2. Características de um Agente. 16

Figura 3. Esquema do Agente de Reflexo Simples (RUSSEL, 2006). 18

Figura 4. Esquema do Agente com Representação do Estado do Mundo (RUSSEL, 2006). 19

Figura 5. Esquema do Agente Baseado em Objetivos (RUSSEL, 2006). 19

Figura 6. Esquema do Agente Baseados na Utilidade (RUSSEL, 2006). 20

Figura 7. Esquema do Agente com Capacidade de Aprendizagem (RUSSEL, 2006). 20

Figura 8. Funções do conjunto Crisp e Fuzzy (SANDRI; CORREA, 1999). 22

Figura 9. Conjuntos fuzzy resultantes das operações de interseção e união. Adaptada de Sandri, Correa (1999). 23

Figura 10. Estrutura de um controlador fuzzy (GONÇALVES, 2001). 24

Figura 11. Robô da categoria de Simulação 3d, do time Kerosin. 26

Figura 12. Robôs da Citzen utilizados na Realidade Mista. 27

Figura 13. Robôs Aibo na categoria 4 patas. 28

Figura 14. Robô Humanóide. 28

Figura 15. Marcadores e linhas do campo que auxiliam nas informações visuais. 31

Figura 16. Modelo de visão do agente. 32

Figura 17. Tela principal de edição do sistema do Xfuzzy. 35

Figura 18. Seleção dos operadores utilizados. 36

Figura 19. Edição dos tipos. 36

Figura 20. Edição da Base de Regras. 37

Figura 21. Tela inicial do Xfuzzy. 38

Figura 22. Conjunto Fuzzy para o eixo Y do atacante. 40

Figura 23. Conjunto Fuzzy para o eixo Y do goleiro. 40

Figura 24. Máquina de Inferência do Primeiro Controlador. 41

Figura 25. Conjunto Fuzzy para o eixo X do atacante e goleiro. 41

Figura 26. Máquina de Inferência do Segundo Controlador. 42

Figura 27. Conjunto Fuzzy para o eixo Y do atacante. 43

Figura 28. Máquina de Inferência do Terceiro Controlador. 43

Figura 29. Conjunto Fuzzy para o eixo Y do atacante. 44

Figura 30. Conjunto Fuzzy para o eixo Y do goleiro. 44

Figura 31. Conjunto Fuzzy para o eixo X do atacante. 45

Figura 32. Conjunto Fuzzy para o eixo X do goleiro. 45

Figura 33. Máquina de Inferência do Quarto Controlador. 45

Figura 34. Regras para o Controlador do eixo Y. 45

Figura 35. Regras para o Controlador do eixo X. 46

Figura 36. Goleiro defendendo o gol e agarrando a bola - contra o Brainstormers. 48

Figura 37. Goleiro impedindo chute a gol - contra o Brainstormers. 49

Figura 38. Bom posicionamento do goleiro, mas falta de marcação – contra o Helios. 49

LISTA DE TABELAS

Tabela 1. Categorias da RoboCup. 29

Tabela 2. Resultados Totais. 47

SUMÁRIO

1 INTRODUÇÃO 10

2 INTELIGÊNCIA ARTIFICIAL 13

2.1 Agentes 15

2.2 Lógica Fuzzy 21

3 FUTEBOL DE ROBÔS 25

3.1 RoboCup 25

3.2 Ambiente de Simulação 30

4 CONTROLADORES DIFUSOS PARA ROBÔS GOLEIROS 34

4.1 Ferramentas Utilizadas 35

4.2 Mudanças no Código Base 38

4.3 Primeiro Controlador Construído 40

4.4 Segundo Controlador Construído 41

4.5 Terceiro Controlador Construído 42

4.6 Quarto Controlador Construído 43

4.7 Resultados Obtidos 46

5 CONSIDERAÇÕES FINAIS 52

REFERÊNCIAS 55

APÊNDICE A - Tabelas com análise dos jogos contra cada time 59

1 INTRODUÇÃO

A presente monografia objetiva descrever uma pesquisa sobre controladores fuzzy para o posicionamento do goleiro em relação ao atacante no futebol de robôs simulado em duas dimensões. Tal pesquisa faz parte do escopo do Bahia Robotics Team – BRT, do qual a autora participa, sendo realizada com fim de melhorar o desempenho do goleiro, mais especificamente, aumentar a quantidade de bolas agarradas pelo mesmo, em detrimento aos gols sofridos, melhorando seu posicionamento para recepcionar a bola.

O BRT é uma união de dois grupos de pesquisa – o Núcleo de Arquitetura de Computadores e Sistemas Operacionais (ACSO/UNEB) e o Grupo de Pesquisa em Computação Inteligente (GPCI/FIB) – visando investigar o estado da arte nas pesquisas em robótica inteligente. Para atingir este objetivo, o BRT optou por dedicar-se ao projeto internacional de pesquisa conhecido como RoboCup, atuando nas categorias de futebol de robôs simulado em duas dimensões e de realidade mista. O BRT possui um time em cada um destas categorias, o Bahia2D e o BahiaMR, respectivamente.

A motivação desta pesquisa foi bem específica, tendo sido inspirada na deficiência do goleiro do time Bahia2D, que ainda sofre muitos gols em relação ao dos demais times classificados para as competições mundiais, levando a crer que sua performance atual é insatisfatória. A quantidade de gols sofridos por um time é influenciada pela zaga e pelo time como um todo, mas o papel fundamental é do goleiro.

No futebol, o papel do goleiro é de extrema importância para a performance do time; se o time não levar gols o pior resultado que pode acontecer é um empate. Assim, fortalecer o goleiro possibilitará ao time Bahia2D alcançar um nível competitivo comparável ao dos campeões mundiais.

A Lógica Fuzzy foi escolhida pois se encaixa muito bem para o futebol, que ocorre em um ambiente dinâmico e impreciso. Como em uma partida não existe uma

definição exata de quando a bola é considerada perto ou longe, muito à esquerda ou apenas à esquerda, rápida ou devagar, a lógica difusa é uma ótima opção para trabalhar com os agentes, pois trata os dados respeitando essa imprecisão. Esta lógica descreve problemas de difícil análise matemática, complexos ou mal-definidos, de maneira aproximada, contudo efetiva.

A pesquisa baseia-se em métodos de estudo qualitativos segundo o enfoque interpretativo, utilizando-se da teoria fundamentada em dados (Grounded Theory). Tal método divide-se em 04 etapas:

- Formulação do Problema: constituiu-se da elaboração e delimitação do problema, incluindo detalhada revisão bibliográfica, estudos aprofundados sobre raciocínio automatizado, lógica fuzzy, redes neurais e computação bioinspirada. Também conduziu-se estudos sobre o funcionamento dos simuladores da Robocup e de outros times já implementados com suas contribuições e limitações.

- Coleta de Dados: após instalação e configuração dos ambientes dos simuladores, elaborou-se o projeto do protótipo do goleiro.

- Análise dos Dados: o produto obtido foi testado empiricamente em competição com jogadores e times projetados por outros grupos de pesquisa. Foram realizadas 20 partidas, com e sem o controlador fuzzy inserido no código do time, as quais foram simuladas contra 5 dos melhores times do mundo e contra o atual campeão brasileiro. Comparou-se os resultados e mediu-se a eficiência do controlador do goleiro. Como o restante do time permaneceu inalterado, a diferença de gols sofridos foi resultado imediato da atuação do goleiro. Outro parâmetro utilizado para validação do controlador foi a quantidade de bolas agarradas pelo goleiro nas duas situações.

- Geração de Conclusões: todos resultados analisados durante a pesquisa estão devidamente registrados nesta monografia.

Neste trabalho pretende-se também demonstrar que os controladores fuzzy construídos podem ser úteis ao time, servindo ao seu propósito de criação.

Os conceitos de Inteligência Artificial necessários para a compreensão desta

monografia - agentes e lógica fuzzy - são apresentados no capítulo 2. Uma explanação sobre o futebol de robôs, a RoboCup e o ambiente de simulação encontra-se no capítulo 3. O capítulo 4 contém a descrição da pesquisa e de sua implementação em si. Por fim, no capítulo 5 os resultados encontrados e as novas metas traçadas serão resumidos nas considerações finais.

2 INTELIGÊNCIA ARTIFICIAL

Para entender o que é Inteligência Artificial (IA) deve-se primeiro saber o que é a Inteligência. Mas esta não é uma tarefa fácil, visto que não existe uma definição única do que ela significa. Vários autores já tentaram definir o que é a Inteligência e até hoje não houve um consenso; todavia, é necessário descrevê-la de algum modo. Uma definição aceita, segundo Rabuske (1995), diz que:

> A Inteligência é o processo contínuo de aquisição, de triagem, de ordenação e de interpretação da informação.

Para entender melhor a Inteligência, novos estudos chegaram a dividí-la em 10 tipos: Lógico-Matemática, Espacial, Lingüística, Pessoal, Comunicativa, Musical, Experimental, Corporal, Contextual e Interna. Para a computação, resolveu-se ater aos aspectos relevantes, tratando-a como uma abstração baseada em comportamentos. Assim, observando-se o comportamento do sistema, resolve-se o problema e faz-se inferências (RABUSKE, 1995).

A Inteligência Artificial tenta imitar a Inteligência em si, seja na hora de pensar ou de agir. Seus objetivos centrais podem ser divididos em: um nível teórico, como a criação de teorias e modelos para a capacidade cognitiva; e, em um nível prático, como a implementação de sistemas computacionais baseados em tais modelos (BITTENCOURT, 1998). Ou seja, a IA visa fazer com que o computador use o conhecimento e raciocínio para desempenhar funções realizadas pelo ser humano (REZENDE, 2005), ela tenta criar agentes inteligentes (RUSSEL, 2006).

Inicialmente, duas linhas de pesquisa destinaram-se à construção de sistemas inteligentes: a Conexionista e a Simbólica.

A linha conexionista propõe a modelagem da inteligência humana através de metáforas de componentes do cérebro: os neurônios e suas ligações sinápticas. A este modelo

matemático dá-se o nome de rede neural. As redes neurais apresentam grande sucesso realizando tarefas de reconhecimento de padrões, como por exemplo sistemas de visão computacional ou modelagem de sistemas complexos como análise de crédito (BARRETO, 1995; BARRETO, 1997).

A linha simbólica baseia-se na lógica e nos sistemas de Post (POST, 1943) e teve McCarthy e Newell como seus principais defensores. Os princípios desta linha foram apresentados por Newell (NAKASHIMA *et al.,* 2004). Na IA simbólica, os sistemas especialistas - programas computacionais capazes de simular o conhecimento de um especialista em um domínio de problema restrito - alcançaram grande destaque na década de 70 consolidando-se como paradigma corrente para o desenvolvimento de sistemas simbólicos inteligentes. Entretanto, estes sistemas apresentam uma visão centralizada do conhecimento, representando um comportamento individual (LOUREIRO, 2002). Nessa mesma linha desenvolveram-se métodos heurísticos e lógicas não convencionais com fim de representar crenças, incoerências e incompletudes, como exemplo a Lógica Fuzzy (BITTENCOURT, 1998).

No final dos anos 70, além da abordagem conexionista descrita anteriormente, surgiram propostas de distribuição da IA simbólica aproveitando o estágio de maturidade alcançado pelas redes de computadores e sistemas distribuídos. Estas propostas deram origem à Inteligência Artificial Distribuída (IAD).

Atualmente, a IAD representa uma das áreas da IA tradicional com maior aplicabilidade a problemas do mundo real (DURFEE, 1991). Esta área dedica-se ao estudo de teorias e implicações práticas a respeito de como reunir um conjunto de agentes computacionais em uma comunidade para solucionar problemas eminentemente distribuídos ou utilizar a robustez das redes de computadores para tratamento de problemas complexos (LOUREIRO, 2002). Por razões históricas, a IAD dividiu-se em dois enfoques: a Solução Distribuída de Problemas (SDP) e os Sistemas Multi-Agentes (SMA).

A SDP concentrou seus esforços no problema a ser solucionado e em como

utilizar os recursos disponíveis nas redes de computadores para tal fim. Os SMA, por outro lado, concentram-se no Agente e em suas propriedades, principalmente naquelas que o levam a cooperar (COHEN, LEVESQUE, 1987; SICHMAN, 1996).

Devido a esta diversidade de paradigmas para o projeto de agentes computacionais inteligentes, a IA deparou-se com uma grande dificuldade em compará-los. A adoção de jogos como problema padrão a ser solucionado permitiu esta comparação.

O Xadrez foi bastante utilizado para comparar diferentes paradigmas, uma vez que este permite tanto a comparação entre dois paradigmas distintos utilizados para implementar a habilidade de um enxadrista humano, bem como confrontar uma dada implementação com um enxadrista também humano. Atualmente, o Xadrez já não é tão desafiador como antes; então, decidiu-se criar um novo desafio: criar agentes inteligentes autônomos jogadores de futebol. Dessa forma, o Futebol passou a ser o problema padrão da IA.

2.1 Agentes

O agente é um ser autônomo capaz de tomar decisões por si mesmo, sem a interferência de um sistema ou de outra entidade. Ele deve ser capaz de interagir com outros agentes e com o ambiente em que se encontra, através de sensores e atuadores, decidindo se deve confiar e cooperar com os demais. Na figura 1 pode-se ver a representação de um agente interegindo com o ambiente.

Na computação, o agente é um software que age para um usuário ou programa. Ele tem a autonomia de decidir se e quando deve executar uma ação, além de se auto convocar para realizar uma tarefa. Na figura 2 pode-se observar algumas características dos agentes.

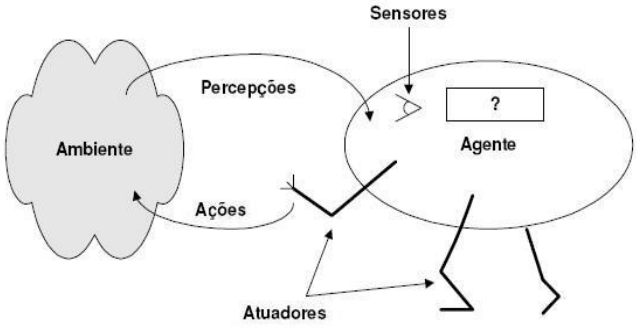

Figura 1. Esquema de um Agente.

Figura 2. Características de um Agente.

O conceito de agente é uma abstração que permite, de forma conveniente e poderesa, descrever uma entidade de software complexa capaz de agir com uma certa autonomia para completar uma tarefa em prol de um usuário. Existem varias derivações deste conceito como:

- **Agentes Inteligentes**

A Inteligência Artificial tem como um ramo projetar agente inteligentes, que devem possuir algumas características básicas, como a habilidade de se adaptar e de aprender. A habilidade adaptativa consiste em explorar o ambiente e se reconfigurar em resposta a ele, o que pode ser atingido através da escolha da alternativa das regras de resolução de problemas ou de algoritmos, ou através da descoberta de estratégias para resolver problemas. A adaptação também pode consistir em outros aspectos internos do agente, como achar recursos de processamento e armazenamento. Já a aprendizagem pode ser através do erro e acerto, o que implica na capacidade de introspecção e análise do

comportamento e sucesso. A aprendizagem também se dá através de exemplos e generalização, o que inplica na capacidade de abstrair e generalizar.

- **Agentes Autônomos**

Agentes autônomos são agentes que capazes de tomar suas próprias decisões, agindo conforme seus objetivos, baseados em sua percepção do ambiente. Mesmo assim, em todas as aplicações importantes eles são surpervisionados por um ser humano, que os coloca para rodar inicialmente, podem modificar seus comportamentos e eventualmente até desligá-los.

- **Agentes Distribuídos**

Os agentes distribuídos são projetados para serem bastante flexíveis, incluindo-se os recursos necessários a ele em sua própria descrição, de forma a facilitar sua execução como threads diferentes em processadores distribuídos.

- **Agentes Móveis**

Agentes móveis são códigos de agentes que se movem para outro processador, mesmo em estado de execução. Eles também são chamados de códigos móveis. Eles podem ser usados para reunir informações, ao se passar de uma máquina para outra fazendo backups. Um vírus pode ser um exemplo de um agente móvel.

- **Sistemas Multi-Agentes**

Sistemas Multi-Agentes é quando inúmeros agentes interagem em um sistema, podendo ter cada um características específicas para agir coletivamente. Normalmente, cada agente não possui todas as informações do meio, tendo que colaborar com seus parceiros para atingir seu objetivo. Às vezes, pode nem haver um controle geral, como é o caso dos sistemas baseados em enxames (swarm systems). Os dados são descentralizados e a execução é assíncrona. Como áreas relacionadas tem-se justamente a IAD e a SDP.

- **Simulação Multi-Agente**

Simulação Multi-Agente vem do princípio dos sistemas multi-agentes voltado à simulação. Seus componentes são dos sistemas multi-agentes só que em um ambiente

simulado. Assim, pode-se estudar melhor os fenômenos emergentes e a dinâmica do sistema. Esse é o caso também do futebol simulado de robôs, onde se encontra os agentes – jogadores – em uma partida simulada de futebol.

É possível classificar os agentes por suas capacidades em cinco tipos distintos, a saber:

- **Agentes de Reflexo Simples (Simple Reflex Agents)**

Possuem um conjunto de regras, de forma que a depender da condição percebida executam determinada ação. Ele apenas reage ao ambiente de acordo com suas regras pré-estabelecidas, como esquematizado na figura 3, sendo considerado o mais simples dos agentes. Eles podem ser comparados aos primeiros sistemas especialistas, diferenciando-se principalmente pela baixa interatividade com o usuário e pela maior autonomia.

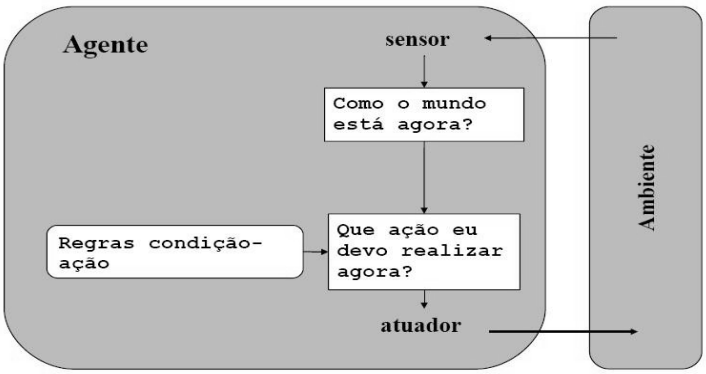

Figura 3. Esquema do Agente de Reflexo Simples (RUSSEL, 2006).

- **Agentes com Representação do Estado do Mundo (Model-Based Reflex Agents)**

Os agentes atualizam dinamicamente sua representação do estado do mundo. Assim, a mesma condição em momentos diferentes pode fazer com que o agente tenha diferentes reações. Ele deve analisar a evolução do mundo, julgando quais as repercursões de suas possíveis ações, e construir uma crença sobre a melhor opção a tomar, como mostrado na figura 4. Esses agentes ainda são classificados como reativos, pois continuam agindo de acordo com regras de condição-reação.

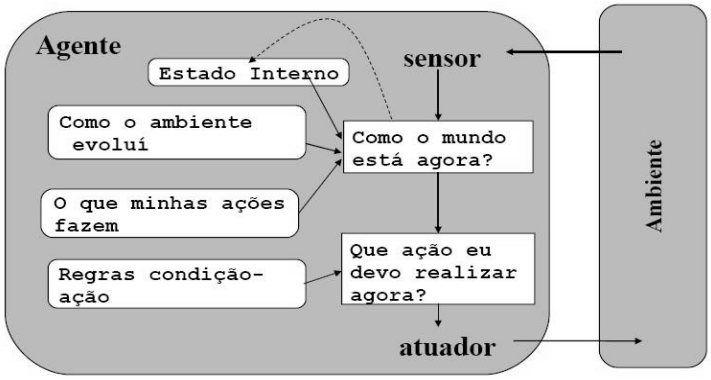

Figura 4. Esquema do Agente com Representação do Estado do Mundo (RUSSEL, 2006).

- **Agentes Baseados em Objetivos (Goal-Based Agents)**

O agente baseado em objetivos, ou metas, leva em consideração seus objetivos para tomar uma decisão sobre qual ação tomar, além do estado atual do ambiente como os agentes com representação do estado do mundo. Ele já é considerado um agente cognitivo, pois tem desejos, crenças e intenções, sendo capaz de um planejamento prévio para que atinja seus objetivos. Assim, uma mesma condição, em um mesmo tempo, pode levar a comportamentos diferentes; deixando-o ainda mais flexível. Na figura 5, pode-se observar um esquema deste tipo de agente.

Figura 5. Esquema do Agente Baseado em Objetivos (RUSSEL, 2006).

- **Agentes Baseados na Utilidade (Utility-Based Agents)**

Este agente estima a satisfação que sentirá ao executar uma ação, relacionada ao

cumprimento de seus objetivos. Quanto maior a satisfação maior as chances dele optar pela ação. Isso ajuda o agente nas situações de incerteza e conflito de objetivo. A figura 6 apresenta um esquema deste tipo de agente interagindo com o ambiente.

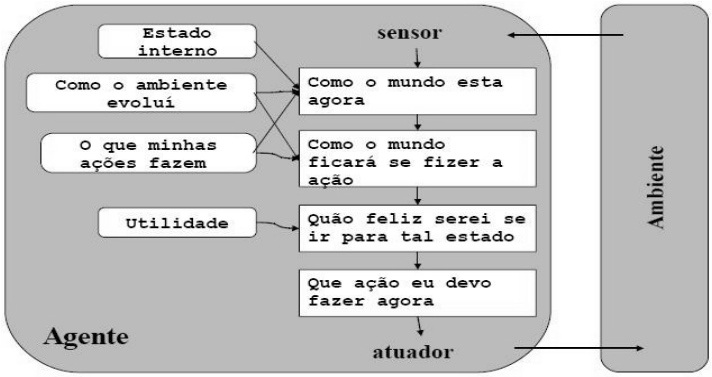

Figura 6. Esquema do Agente Baseados na Utilidade (RUSSEL, 2006).

- **Agentes com Capacidade de Aprendizagem (Learning Agents)**

Os agentes aprendem e se adaptam a novas situações com base em suas experiências. Ele precisa analisar seu comportamento e o grau de sucesso relacionado, para poder escolher qual as melhores escolhas tomadas e quais deve evitar em determinado caso. Ele deve ser capaz de se adaptar em tempo real e de aprender rapidamente através de grandes bases de dados, além de possuir uma memória de curto e longo prazo. O agente também pode aprender através de exemplos, observando outros agentes, ou pela generalização. Para entendê-lo melhor, a figura 7 apresenta um esquema deste tipo de agente interagindo com o ambiente.

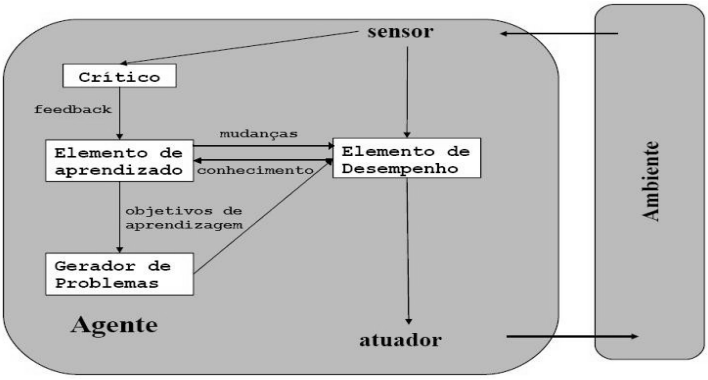

Figura 7. Esquema do Agente com Capacidade de Aprendizagem (RUSSEL, 2006).

2.2 Lógica Fuzzy

A Lógica Fuzzy é uma teoria derivada da lógica booleana - lógica nítida - e desenvolvida especialmente para representação do conhecimento humano, que é impreciso, vago, incerto, difuso. Daí o nome *fuzzy*.

Os sistemas de computadores fuzzy, diferentemente dos outros sistemas mais tradicionais, não classificam uma informação apenas como falsa ou como verdadeira; não há uma resposta simples, como um sim ou não. Para eles existe um meio termo, como um pouco, razoável ou muito, permitindo a pertinência parcial de um elemento a um conjunto.

A Teoria dos Conjuntos Fuzzy foi criada em 1965 por L. A. Zadeh, professor da Universidade da Califórnia, em Beckerley, nos Estados Unidos.

A Lógica Fuzzy é algumas vezes considerada controversa e não é aceita como uma verdadeira lógica em certos círculos científicos, visto que seus modelos permitem soluções aproximadas que não correspondem à uma "verdade" lógica. Todavia, ela é cada vez mais utilizada na robótica e em aparelhos domésticos.

Para construir um modelo fuzzy associa-se um valor $\mu(p)$ a uma proposição p, indicando o grau de veracidade desta proposição. A Imagem de $\mu(p)$ está entre 0 e 1. Se p for verdade associa-se ele a 1 e se for falso a 0, como na lógica booleana; entretanto, tem-se agora infinitas outras opções de pertinência.

- $\mu(p) = 1$, se verdadeiro
- $\mu(p) = 0$, se falso
- $0 < \mu(p) < 1$, meio termo

Este terceiro exemplo é o que diferenciou a Lógica Fuzzy, tornando-a mais flexível. A variação de p é chamada de **universo de discurso**; por exemplo, se o valor máximo de p for igual a 10 e o mínimo for -10, então o universo de discurso de p varia entre -10 à 10. Na figura 8. tem-se um exemplo representando a lógica clássica e a fuzzy: na

clássica (conjunto crisp) uma pessoa ou é um adolescente ou não é, definindo-se uma idade para a transição; na fuzzy existe uma transição onde a criança vai entrando na adolescência gradativamente e, da mesma forma, vai saindo dela. No caso, o universo de discurso do conjunto fuzzy varia de 11 à 19.

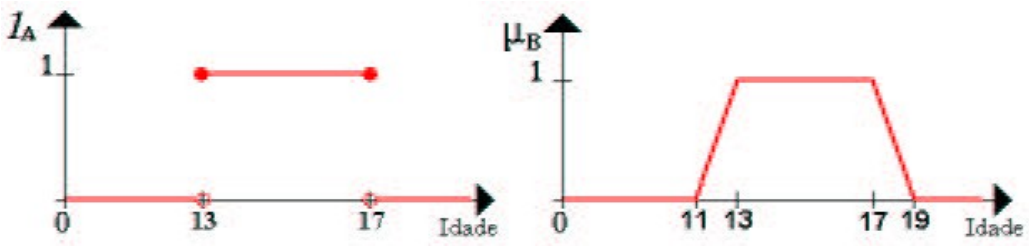

Figura 8. Funções do conjunto Crisp e Fuzzy (SANDRI; CORREA, 1999).

A Lógica Fuzzy possui também os **operadores** clássicos E, OU e NAO, adaptados às suas peculiaridades. Várias funções podem estar associadas ao E-fuzzy, OU-fuzzy e NAO-fuzzy. As utilizadas neste trabalho são:

- E-fuzzy (x, y) = mínimo (x, y)
- OU-fuzzy (x, y) = máximo (x, y)
- Não-fuzzy (x) = 1 - x

Para definir a operação de união (OU-fuzzy) entre conjuntos fuzzy, utiliza-se uma família de funções, as normas S, que possuem as seguintes propriedades (REZENDE, 2005):

- Comutatividade: S(a,b) = S(b,a)
- Associatividade: S(a,S(b,c)) = S(S(a,b),c)
- Monotonicidade: se a <= b e c <= d, então S(a,c) <= S(b,d)
- Coerência de contornos: S(a,1) = 1 e S(a,0) = a

Para a operação de interseção (E-fuzzy) ocorre o mesmo, sendo as propriedades (REZENDE, 2005):

- Comutatividade: S(a,b) = S(b,a)
- Associatividade: S(a,S(b,c)) = S(S(a,b),c)

- Monotonicidade: se a <= b e c <= d, então S(a,c) <= S(b,d)
- Coerência de contornos: S(a,1) = a e S(a,0) = 0

Na figura 9, pode-se ver um exemplo de como essas operações funcionam, através do enfoque de Zadeh.

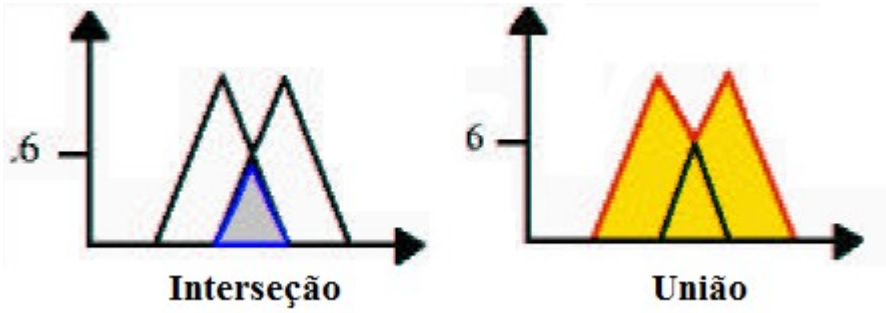

Figura 9. Conjuntos fuzzy resultantes das operações de interseção e união.
Adaptada de Sandri, Correa (1999).

Segundo Rabuske (2005), a capacidade da representação fuzzy do conhecimento de classificar de forma imprecisa as variáveis de um problema, de forma qualitativa ao invés de quantitativa, representa a idéia de uma **variável linguística**. Esta é utilizada para representar um conceito ou variável de forma imprecisa, logo linguística, admitindo apenas expressões linguísticas como valores. Algumas das variáveis linguísticas usadas neste trabalho foram: mtolonge, longe, medio, perto, mtoperto, mtoesq, esq, meio, dir e mtodir.

Para representar a base de conhecimento fuzzy utiliza-se as **regras de produção,** que em sua forma mais conhecida, de Zadeh-Mandani, é composta de duas partes principais:

se <antecedente> então <consequente>

Ou em inglês:

if <premise> then <consequence>

A cada regra pode-se associar também um grau de confiança, ou peso da regra. Assim, pode-se fazer com que uma regra seja mais relevante que outra.

Para realizar a **inferência**, utiliza-se justamente esssas regras para mapear as variáveis de entrada com as de saída. As variáveis de entrada sofrem *fuzificação,* que consiste

em encontrar o grau de pertinência delas no conjunto fuzzy, transformando cada uma em um termo nebuloso. Já as variáveis de saída sofrem *defuzificação*, processo onde é calculado um valor numérico para elas, baseado na inferência de seu grau de pertinência (KASABOV, 1999).

O método de defuzificação utilizado neste trabalho foi o de centro de área, que encontra o centro geométrico do universo de discurso da variável de saída.

Na figura 10, pode-se visualizar como funciona um controlador difuso e sua relação com o ambiente.

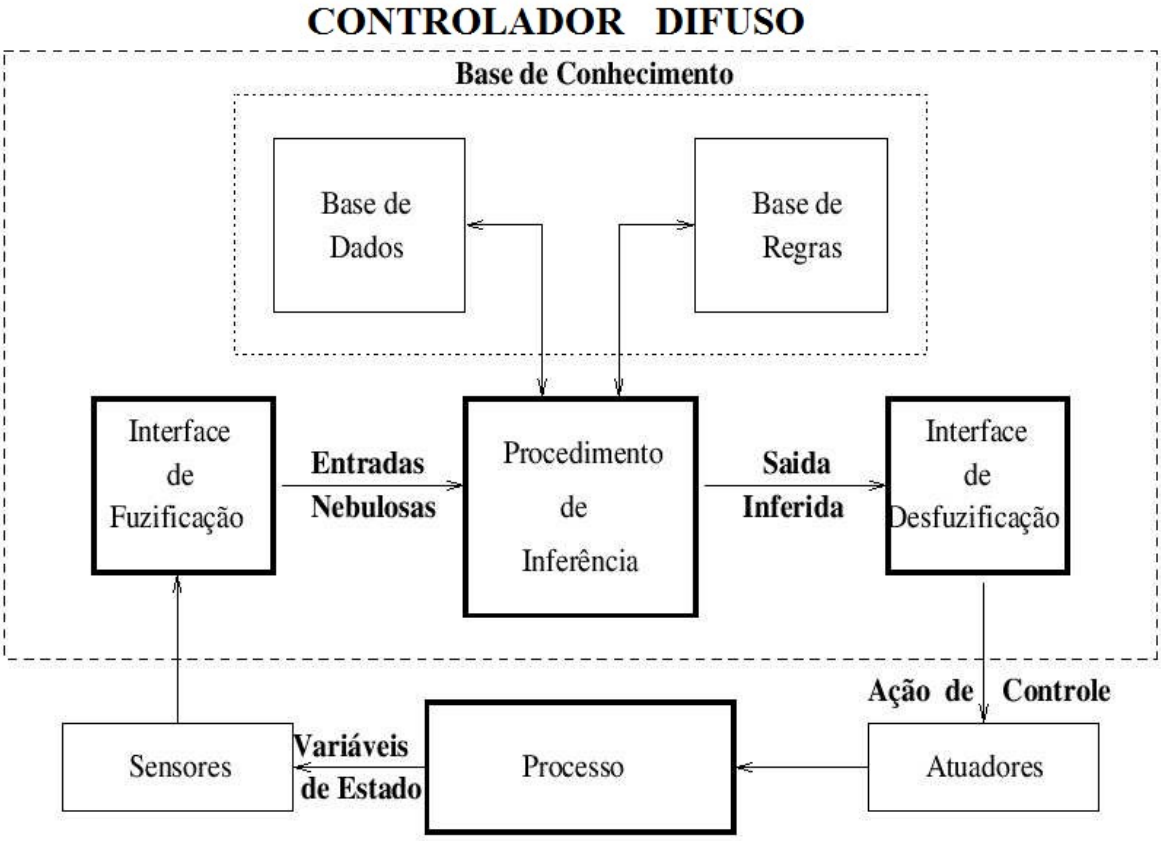

Figura 10. Estrutura de um controlador fuzzy (GONÇALVES, 2001).

3 FUTEBOL DE ROBÔS

A exemplo do xadrez, em 1996, um grupo internacional de pesquisadores em Inteligência Artificial e Robótica Inteligente propôs um desafio: uma partida de futebol entre robôs autônomos. O futebol de robôs reúne grande parte dos desafios presentes em problemas eminentemente distribuídos do mundo real, tais como, veículos autônomos, busca de informação em bases de dados distribuídas, planejamento da geração de energia elétrica, recomposição de linhas de transmissão, controle de tráfego aéreo e urbano, etc. (LOUREIRO, 2002) Sendo assim, o futebol de robôs apresenta-se como um laboratório para pesquisa e ensino em automação e informática industrial.

3.1 Robocup

Da iniciativa de utilizar o futebol como problema padrão da IA surgiram duas ligas: a RoboCup (Robot World Cup) Federation e a FIRA (Federation of International Robot-soccer Association). Estas ligas se diferem basicamente nas condições de contorno do problema proposto. A FIRA propôs inicialmente uma partida de futebol composta por times de três robôs cada chamada MiroSot. Um computador ligado a uma câmera de vídeo, localizada dois metros acima do campo, que fornece uma vista do topo do campo e da posição dos robôs, é utilizado para controlar remotamente os robôs que compõem o time.

Outras categorias de competição foram adicionadas posteriormente: NanoSot - times com cinco robôs; SimuroSot - robôs simulados; HuroSot - robôs humanóides com 40 cm de altura e K-pheraSot. Esta proposta da FIRA favorece às soluções centralizadas onde os

robôs possuem baixo grau de autonomia.

A RoboCup propôs inicialmente três diferentes condições de contorno para o problema do futebol de robôs, organizadas em três categorias distintas: robôs simulados, robôs de pequeno porte e robôs de médio porte.

A categoria de robôs simulados sudividiu-se em três: a de duas dimensões, a de três dimensões e a realidade mista.

A categoria de três dimensões possui um simulador bastante novo e depara-se atualmente com diversos problemas a serem contornados. No momento, a partida é realizada dois contra dois, e os robôs mal conseguem se manter em pé. Na figura 11, pode-se ver um robô andando.

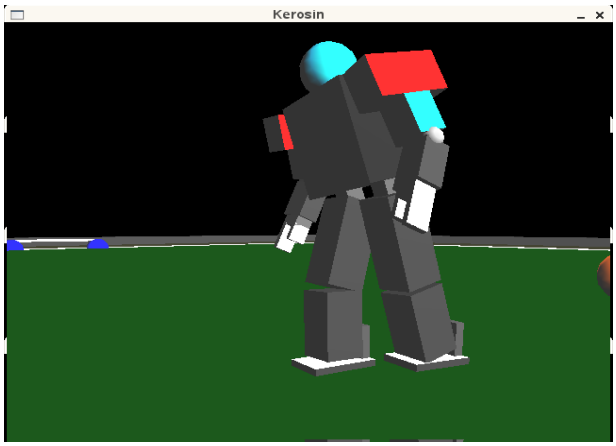

Figura 11. Robô da categoria de Simulação 3d, do time Kerosin.

A realidade mista (RM) é a mais nova categoria, ela mistura – como o nome sugere – a realidade virtual com a realidade física. Tem-se dois pequenos robôs físicos de cada time que jogam futebol em um campo virtual projetado por um monitor posicionado na horizontal. A bola também é virtual. O sistema de visão dos jogadores é global, e se dá através de uma câmera localizada acima do campo. Os robôs recebem os comandos do servidor através de ondas infra-vermelhas, transmitidas por emissores que se encontram localizados nas proximidades do campo. É uma categoria bastante promissora, com várias propostas de

novas aplicações. Na figura 12, pode-se ver os robôs no campo de futebol virtual.

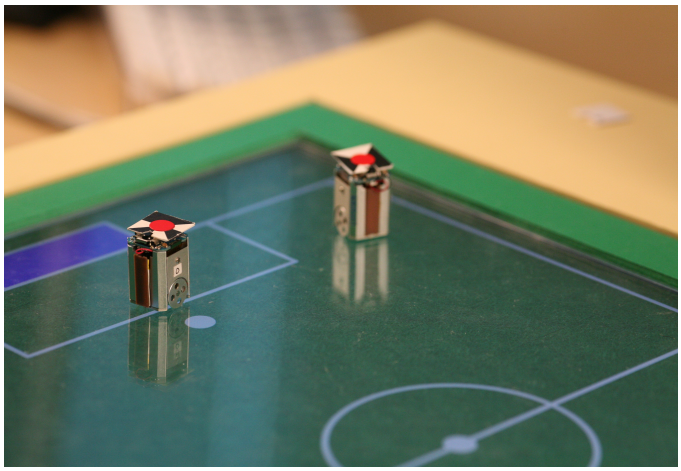

Figura 12. Robôs da Citzen utilizados na Realidade Mista.

Na categoria de robôs simulados em duas dimensões os times possuem onze robôs, como no futebol de campo; já nas outras duas categorias originais cada time possui cinco jogadores.

Diferentemente da Fira, na RoboCup os robôs são completamente autônomos nas categorias de robôs simulados e de médio porte. Na categoria de robôs de pequeno porte, ainda é permitido a utilização de um sistema de visão global, onde a imagem de câmera é processada e posteriormente transmitida para os robôs, como na realidade mista. Existem ainda na categoria de robôs de pequeno porte, times que utilizam o sistema de visão embarcado.

Com o decorrer do tempo, novas categorias foram sendo adicionadas e outras modificadas.

A categoria de robôs com quatro patas, na qual o AIBO (robô cachorro desenvolvido pela Sony) era utilizado, está em fase de substituição de seus robôs padrão, visto que a Sony descontinuou o produto. A figura 13 mostra dois robôs cachorros jogando futebol, sendo que o robô vermelho está prestes a cobrar um pênalti contra o do time azul.

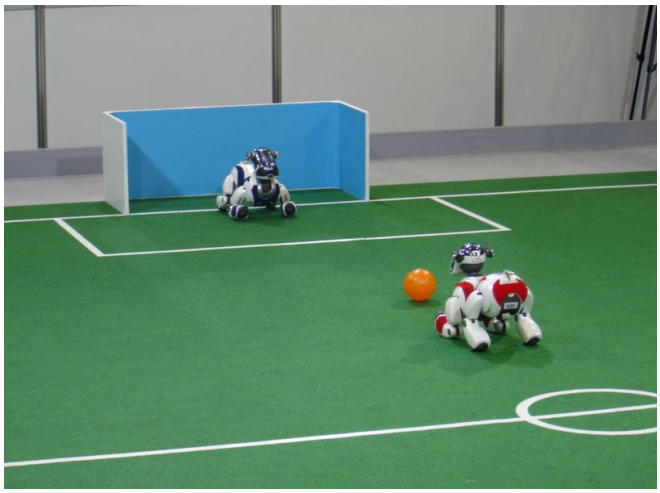

Figura 13. Robôs Aibo na categoria 4 patas.

A categoria de robôs humanóides divide-se em KidSize, com robôs de 30 a 60 cm de altura, e TeenSize com robôs um pouco maiores, de 80 a 130 cm de altura. Essa categoria, como a de pequeno e médio porte, pode utilizar robôs comprados prontos ou construídos pela própria equipe, de forma que siga as especificações da liga. A figura 14 mostra um dos robôs humanóides da equipe Nimbro, da Universidade de Freiburg na Alemanha, atual campeã mundial da TeenSize.

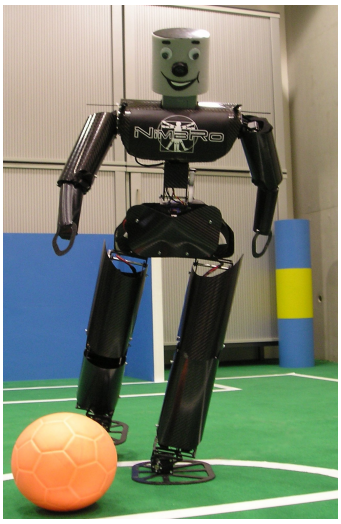

Figura 14. Robô Humanóide.

Para abranger novos problemas, outras modalidades também foram criadas. Como, por exemplo, a RoboCupRescue, onde o problema em questão é o resgate em

escombros realizados por robôs autônomos. Para fomentar o interesse pela robótica em alunos do ensino médio e fundamental, criou-se ainda a RobocupJunior, com fins mais educacionais. A Robocup@Home também é uma novidade, incentivando o desenvolvimento de robôs para solucionar tarefas do cotidiano doméstico.

Na tabela 1, pode-se ver uma lista das categorias atuais da RoboCup e algumas de suas sub-ligas.

Tabela 1. Categorias da RoboCup.

RobocupSoccer		
	Simulação	
		2D - Duas Dimensões
		3D - Três Dimensões
		RM - Realidade Mista
	Robôs Pequenos	
	Robôs Médios	
	Robôs com 4 patas	
	Robôs Humanóides	
RobocupRescue		
Robocup@Home		
RobocupJunior		

O futebol de robôs no formato proposto pela RoboCup Federation, particularmente em sua categoria de robôs simulados, onde os robôs apresentam um grau de autonomia maior, é o objeto de estudo desta pesquisa.

Anualmente, a RoboCup Federation promove competições de todas as categorias em paralelo a um Simpósio Internacional onde as contribuições científicas de cada equipe são apresentadas e publicadas. Em 2004, os times utilizaram abordagens variadas e nenhuma delas se sobressaiu largamente sobre as demais (LIMA *et al.*, 2005). A equipe russa STEP venceu a competição utilizando um modelo de raciocínio baseado em regras descrevendo cenários e condições (STANKEVICH, 2004). A equipe alemã Brainstormers, terceira colocada na época e atual campeã mundial, utilizou aprendizagem por reforço para os

diversos comportamentos desejados (RIEDMILLER, 2005). Também foram utilizados métodos evolucionários (NAKASHIMA *et al.*, 2004) e contribuições em técnicas para filtragem dos ruídos nas percepções recebidas pelos robôs e estratégias para modelagem de times (GONÇALVES, LOUREIRO, BITTENCOURT, 2001). No Brasil, alguns times utilizam a Lógica Fuzzy. O Bahia2D começou com controladores fuzzy apenas para os atacantes (SILVA, SIMÕES, ARAGÃO, 2007) e, com o desenvolvimento da pesquisa, esses controladores foram extendidos para o restante dos jogadores (SILVA *et al.*, 2007), fazendo com que o time se classificasse para a RoboCup Internacional 2007 e conseguisse a terceira colocação na RoboCup Brasil 2007.

3.2 Ambiente de Simulação

O futebol de robôs simulado da RoboCup ocorre no simulador SoccerServer. Este cria um ambiente virtual e simula a física, a bola e os jogadores, sendo assim possível despreocupar-se dos problemas técnicos e concentrar-se apenas nos conceitos de alto nível, como a inteligência dos jogadores.

O SoccerServer é um sistema que permite agentes autônomos, escritos em diversas linguagens, jogarem uma partida de futebol. São dois times, com onze jogadores cada, que jogam um contra o outro em um ambiente bidimensional. Cada time também possui um técnico. Para isso, existe uma conexão UDP entre os agentes e o SoccerServer, no modelo de cliente-servidor. O servidor cria o campo de futebol e simula os movimentos da bola e dos jogadores, administrando o jogo através de regras pré-estabelecidas; enquanto cada cliente controla os movimentos de um jogador. Os jogadores enviam seus pedidos referentes às ações que desejam executar ao servidor e este, por sua vez, trata os pedidos e atualiza o mundo.

Um agente recebe informações do mundo através da visão e audição. No início de

cada ciclo o servidor envia os dados do corpo para cada agente, como sua reserva de energia e velocidade. O jogo dura dois tempos de 3000 ciclos cada, sendo que um ciclo corresponde a 100ms. Os ciclos são intervalos de tempo discretos utilizados pelo servidor, que é um sistema de tempo real. Se um jogador tiver uma performance lenta ele pode acabar perdendo a oportunidade de agir, pois as ações a serem executadas em um ciclo precisam chegar ao servidor em um certo intervalo de tempo.

As informações visuais consistem de cada objeto no campo de visão do agente, como a bola, linhas, marcadores (flags) e até mesmo outros agentes. Todas as informações são relativas ao agente em si, de forma que ele precisa saber as exatas coordenadas dos marcadores para estimar a posição dos demais objetos, incluindo a dele próprio. No total, são 55 marcadores imaginários no campo, como pode-se ver na Figura 15.

As informações referentes aos marcadores são dadas em coordenadas cartesianas. Adotou-se como origem (0, 0) do plano cartesiano o centro do campo (*flag* c). O eixo x é positivo em direção a *linha b*, para baixo; enquanto o eixo y é positivo em direção a *linha r*, para direita.

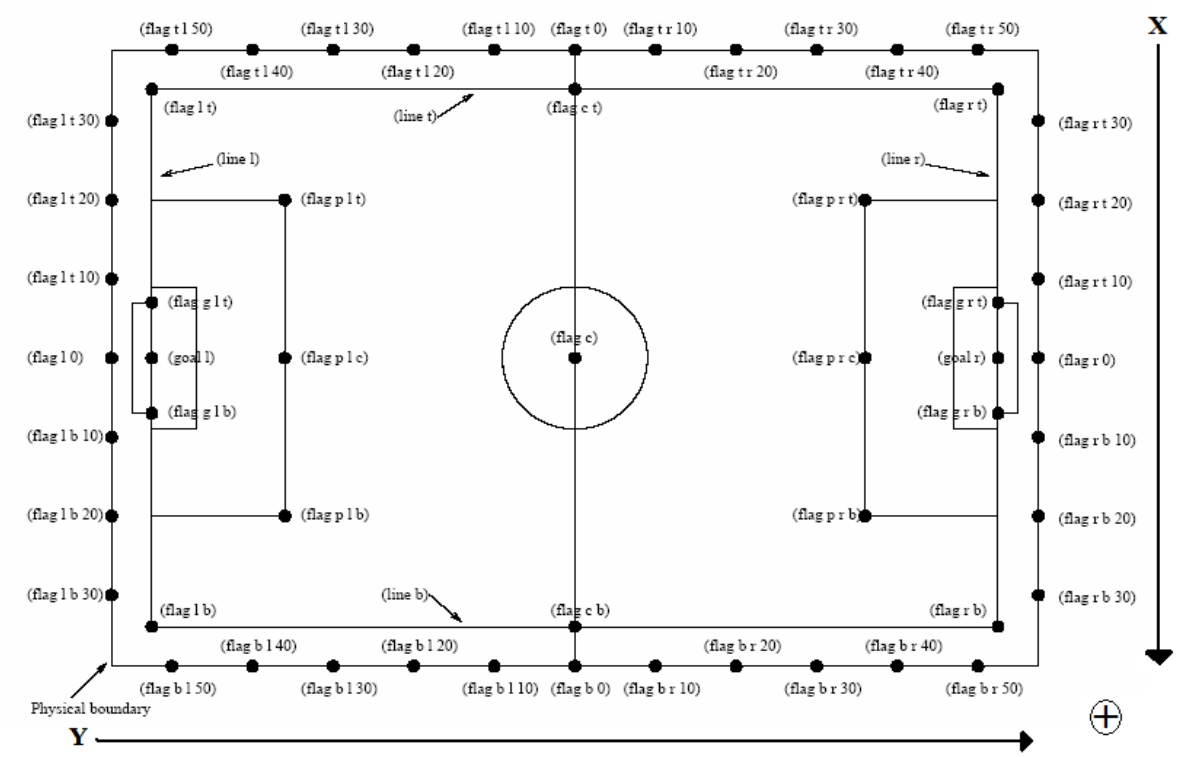

Figura 15. Marcadores e linhas do campo que auxiliam nas informações visuais.

As demais informações visuais são fornecidas em coordenadas polares. Quando um objeto está em seu cone de visão (figura 16), o agente recebe a distância do objeto e o ângulo formado com seu centro de massa. Quando o objeto está à esquerda o ângulo possui valores negativos e à direita valores positivos.

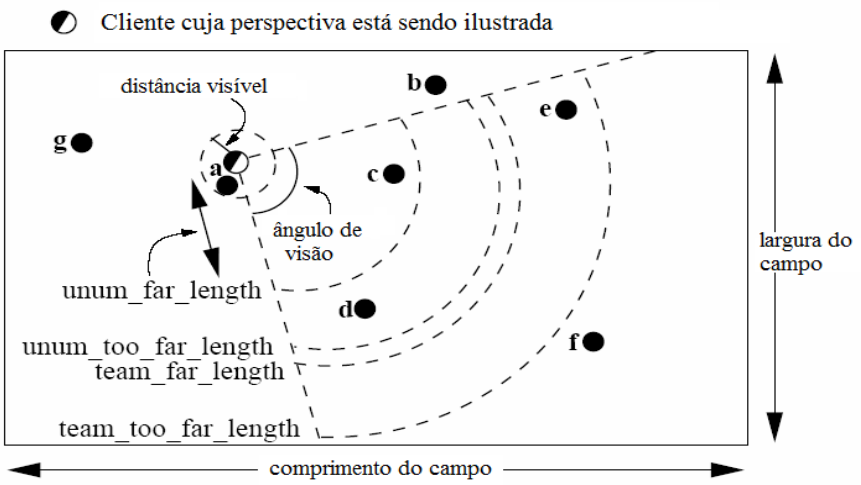

Figura 16. Modelo de visão do agente.

As informações auditivas podem ser provenientes de outros jogadores ou de mudanças na situação do jogo, como um gol por exemplo. As conversas entre os jogadores devem passar pelo servidor, e devem conter no máximo dez signos, de forma que é possível enviar 60 bits de conversa por ciclo. A notícia é enviada no início do ciclo seguinte, podendo ser captada por um jogador com até 50 metros de distância do emissor.

O SoccerServer consiste na realidade de dois programas: o **soccerserver** e o **soccermonitor**. O servidor em si é responsável pela simulação da partida; enquanto o monitor é o programa que mostra o campo virtual criado pelo servidor na tela, utilizando o sistema X window do Linux. É possível conectar inúmeros monitores a um único servidor, para que a partida seja exibida em várias telas.

Existe também uma ferramenta bastante útil que é o **logplayer**. Ela pode ser considerada como um *video player*, pois pode ser usada para rever um jogo através do log

gravado. Isso é possível pois o servidor fornece uma opção onde se pode gravar todos os dados de um jogo para futuras consultas.

Para controlar a partida existem regras aplicadas automaticamente por um juiz virtual, ou por um juiz humano. Algumas das regras aplicadas pelo árbitro automático são: Kick-Off, Goal, Out of Field, Player Clearance, Play-Mode Control, Offside, Backpasses, Free Kick Faults, Half-Time e Time-Up. Pode-se observar que os nome das regras são bastante intuitivos, vindos do futebol real. Já as regras aplicadas pelo árbitro humano estão ligadas a falta de *fairplay*, que é o jogo limpo, pois analisar a intenção de um jogador em uma jogada é bastante subjetivo e portanto mais difícil de ser analisada automaticamente.

4 CONTROLADORES DIFUSOS PARA ROBÔS GOLEIROS

A idéia de utilizar controladores difusos para o futebol de robôs simulado não é nova, algumas outras equipe já publicaram artigos sobre o assunto, como a exemplo do time baiano da Universidade Federal da Bahia, o MecaTeam.

O Bahia2D começou a explorar essa técnica no início de sua formação. Cada membro ficou responsável por criar controladores difusos para determinado tipo de jogador, a fim de melhorar sua performance. A autora ficou incumbida especificamente do goleiro e assim teve início esta pesquisa.

Alguns membros da equipe se graduaram e defenderam seus trabalhos de conclusão de curso sobre seus controladores, como os trabalhos de Silva (2007), Pereira (2007) e Cruz (2007).

Este trabalho propõe um controlador difuso para o posicionamento sem bola do goleiro, em relação ao atacante. Sem bola porque ele visa que o goleiro se posicione na melhor posição justamente para a recepção da mesma. E em relação ao atacante pois o goleiro leva em consideração apenas a posição do atacante com a posse de bola; uma vez chutada a bola a gol, o goleiro já deve estar na posição mais provável para recpcioná-la.

Para chegar ao controlador final deste trabalho, várias versões precisaram ser construídas e testadas. A fase de testes e ajustes foi a que consumiu maior tempo. As principais versões construídas serão apresentadas neste capítulo.

Um outro desafio foi achar o parâmentro de entrada do controlador, visto que no modelo de mundo do time Bahia2D não existia uma função para retornar a posição do jogador com a posse de bola. Essa função precisou ser criada no modelo de mundo e ajustada no código do goleiro. Ela será mostrada também neste capítulo.

Para chamar o controlador fuzzy foi criada uma função *melhorPosicaoGoleiro()*, que se encarrega de passar os parâmetros corretos para o controlador e retornar a posição em

coordenadas cartesianas na qual o goleiro deve se posicionar. Mais adiante, neste capítulo, entrar-se-á em maiores detalhes sobre esta função.

4.1 Ferramentas Utilizadas

Para construir os controladores foi utilizado a ferramenta gráfica Xfuzzy, em sua versão 3.0.0. Este programa possibilita a criação da máquinas de inferências de forma intuitiva e prática e, uma vez montadas, pode-se gerar o código em C, C++ ou Java.

Na figura 17 pode-se ver a tela principal de edição de sistema deste ambiente.

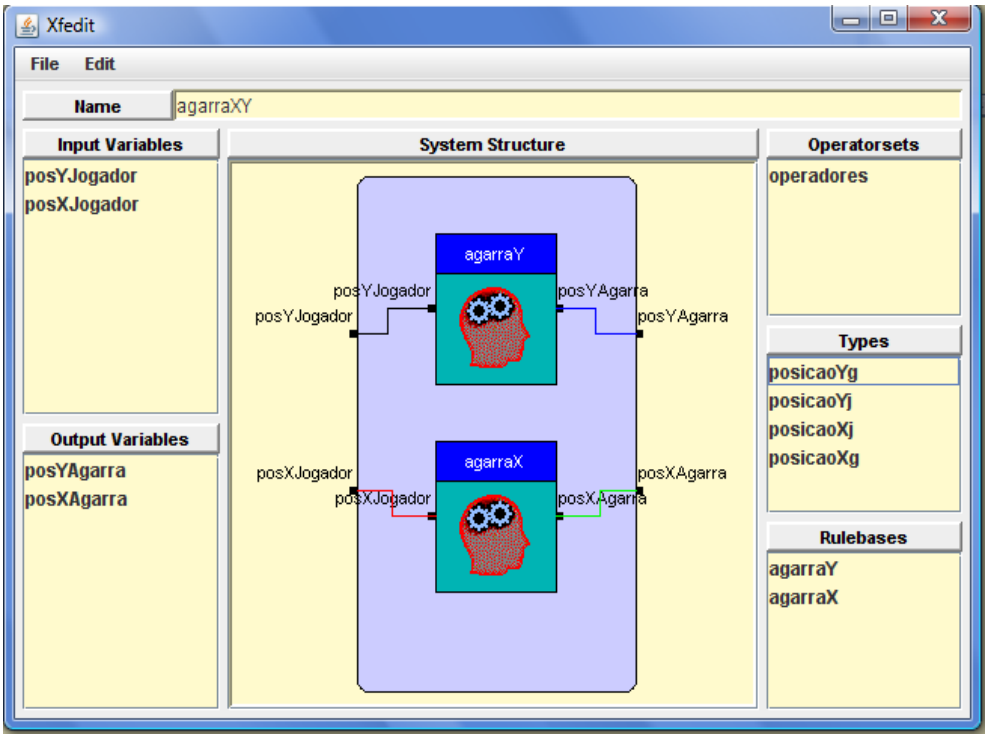

Figura 17. Tela principal de edição do sistema do Xfuzzy.

Primeiramente, deve-se especificar os operadores fuzzy, no caso utilizou-se as opções indicadas na figura 18.

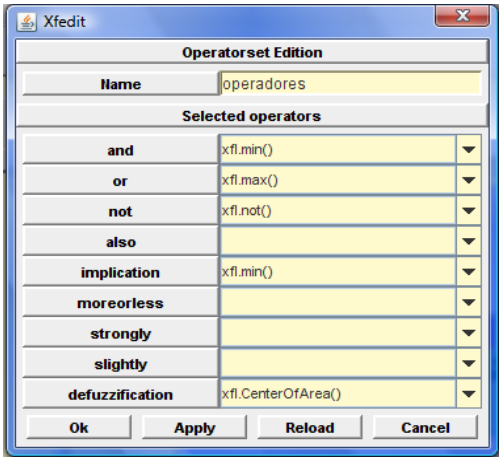

Figura 18. Seleção dos operadores utilizados.

Depois, deve-se criar os tipos. Abre-se uma outra janela para editá-lo, onde define-se seu nome, o universo de discurso e as funções de pertinência. Na figura 19 pode-se ver esta janela de edição. Deve-se escolher o tipo de função a ser utilizada também, no presente trabalho usou-se a trapeizodal.

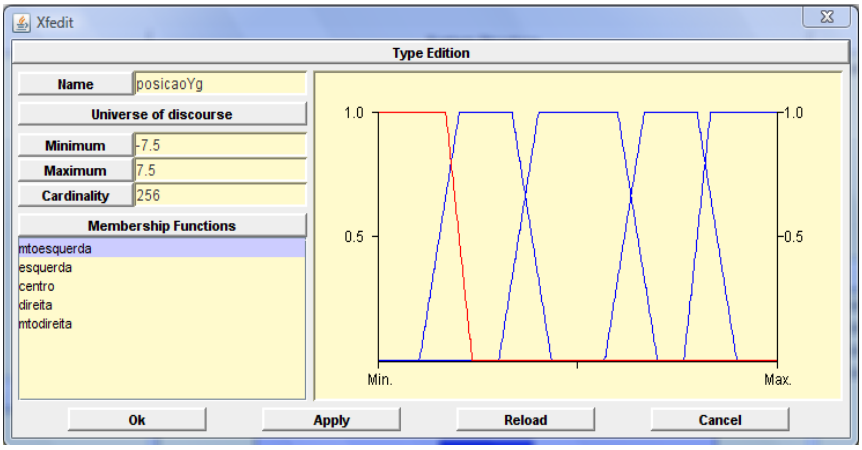

Figura 19. Edição dos tipos.

Em seguida, cria-se as variáveis de entrada e saída, definindo-se o tipo de cada uma.

Com essas informações definidas, falta criar a base de regras, como ilustrado na figura 20. Deve-se definir o nome, o conjunto de operadores a ser utilizado, as variáveis de

entrada e saída e as regras em si. Para isso, pode-se escolher entre três formas de edição diferentes: uma forma livre (como indicado na figura 20), uma forma de tabela ou uma forma de matriz. Utilizou-se tanto a forma de tabela como a livre. Nesta última, deve-se definir o peso da regra; a condição, escolhendo-se graficamente as variáveis e os operadores envolvidos; e a conclusão. É possível, também, digitar as regras diretamente no código do arquivo, através de um editor de textos; o que também foi utilizado, por conceder uma maior liberdade.

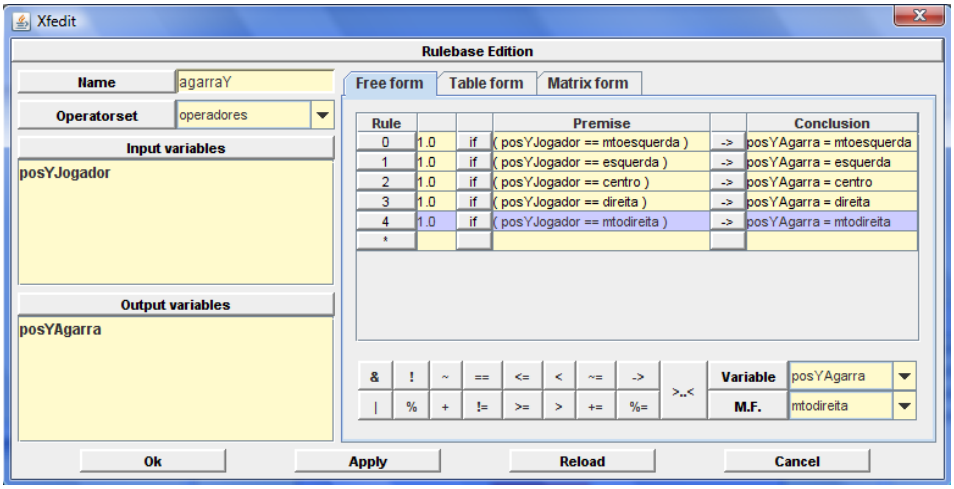

Figura 20. Edição da Base de Regras.

Agora, falta apenas inserir a máquina de inferência na estrutura do sistema, escolhendo-se a base de regras relacionadas. As variáveis de entrada e saída são dispostas automaticamente nessa região gráfica, como mostrado na figura 17. Um vez inserida a máquina, falta apenas ligar as variáveis de entrada e saída desejadas.

Salva-se o trabalho e, na primeira tela do Xfuzzy, mostrada na figura 21, manda-se compilar em uma das linguagens disponíveis.

Apesar do código do time ser predominantemente em C++, a geração do código fuzzy nesta linguagem não ficou compatível; assim, optou-se por gerar o código em C, que funcionou com apenas alguns ajustes, como correções de certas palavras reservadas e especificações de parâmetros em algumas funções.

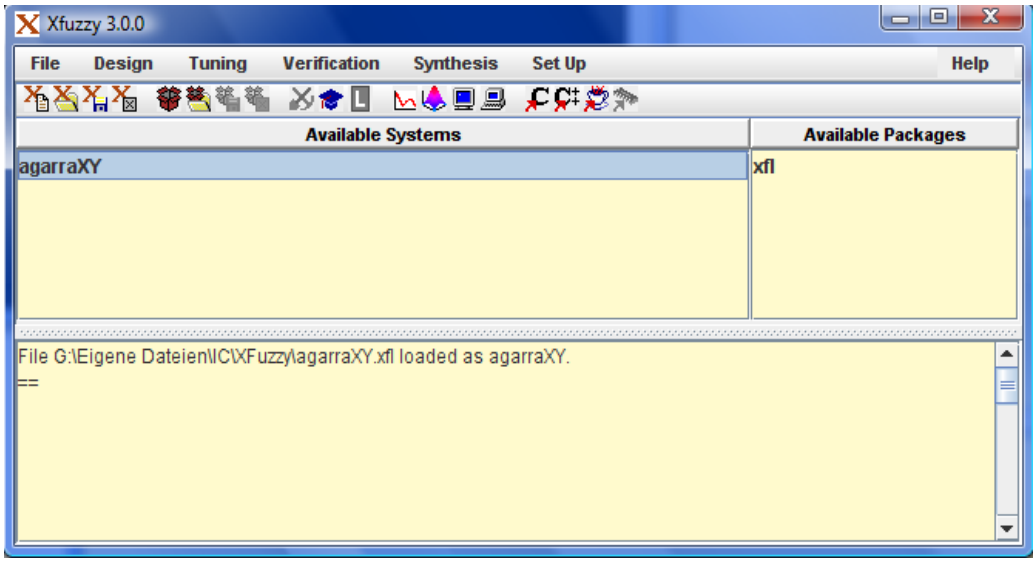

Figura 21. Tela inicial do Xfuzzy.

O Xfuzzy também oferece opções de verificação, onde se pode monitorar visualmente o comportamento do controlador. Ferramenta bastante útil para ajustes nas funções de pertinência. Pode-se também gerar gráficos do sistema em duas ou três dimensões e rodar simulações dele.

4.2 Mudanças no Código Base

A função *posJogadorPosseBola()* foi criada e inserida no modelo de mundo do time. Ela utilizou como auxílio uma função já existente que retornava o objeto mais rápido em relação a outro – *getFastestInSetTo()*. Foram passados como parâmetro os jogadores e a bola, assim ela retornaria o jogador mais rápido da bola - *getFastestInSetTo(OBJECT_SET_PLAYERS, OBJECT_BALL, &iCyc)*. Utilizou-se também uma função que retorna a posição de um objeto – *getGlobalPosition()* – para achar a posição do jogador mais rápido da bola. A função final ficou assim:

```
VecPosition WorldModel::posJogadorPosseBola()
{
  int iCyc;
  ObjectT o = getFastestInSetTo( OBJECT_SET_PLAYERS, OBJECT_BALL, &iCyc );
  return getGlobalPosition(o);
}
```

Esta função apenas retorna a posição cartesiana do jogador mais rápido da bola. Ela é tratada no código do goleiro, de forma que retorne a posição do jogador adversário com posse de bola. Para isso, criou-se uma condição em que ela é apenas chamada caso a posse de bola não seja do time do goleiro. Assim, garante-se que o retorno dela é em relação a um jogador adversário. A situação foi mais complicada para garantir a posse de bola, precisou-se definir uma distância da bola ao jogador em questão para ser considerado que ele detenha a posse de bola. Essa definição foi empírica e baseada em vários testes, mas estudos já estão sendo levantados para que no futuro encontre-se uma distância ótima para tal. Uma vez definida essa distância, criou-se uma condição para testar se o jogador adversário mais rápido até a bola realmente possui a posse da mesma; caso positivo, o controlador fuzzy entra em ação e, em caso negativo, o goleiro mantém seu comportamento padrão.

Para facilitar a chamada do controlador fuzzy, foi inserido no código do goleiro uma função *melhorPosicaoGoleiro()*, que chama a máquina de inferência com os parâmetros corretos e retorna a posição na qual o goleiro deve se posicionar, como dito anteriormente. O código desta função pode ser visto a seguir:

```
VecPosition Player::melhorPosicaoGoleiro()
{
        double posicaoY;
        double posicaoX;
        agarraXYInferenceEngine(WM->posJogadorPosseBola().getY(),
        WM->posJogadorPosseBola().getX(),&posicaoY,&posicaoX);
        return VecPosition(posicaoX,posicaoY);
}
```

4.3 Primeiro Controlador Construído

O primeiro controlador criado foi simples, ele dizia respeito ao posicionamento do goleiro no eixo Y apenas, ou seja, o movimento de um lado para o outro.

Os parâmetros de entrada eram as posições Y do atacante e do goleiro. A saída era a posição Y que o goleiro deveria assumir para defender o gol.

Por exemplo, se o atacante viesse pela direita e o goleiro estivesse muito a direita, o goleiro seguiria a lógica de que o atacante evitaria chutar em cima dele e se movimentaria para a esquerda, prevendo o local do possível chute.

O maior problema observado com esse controlador foi o posicionamento do goleiro no eixo X, pois diversas vezes ele ficava adiantado demais para agarrar a bola e acabava por tomar gols pelas costas.

As variáveis linguísticas para o atacante foram **esq, meio** e **dir,** apresentadas na figura 22**.** Enquanto para o goleiro foram **mtoesq, esq, meio, dir** e **mtodir,** figura 23**.** A máquina de inferência para este controlador está representada na figura 24.

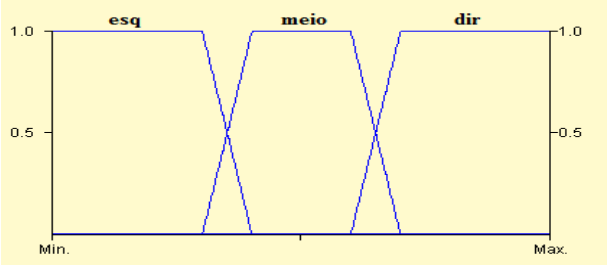

Figura 22. Conjunto Fuzzy para o eixo Y do atacante.

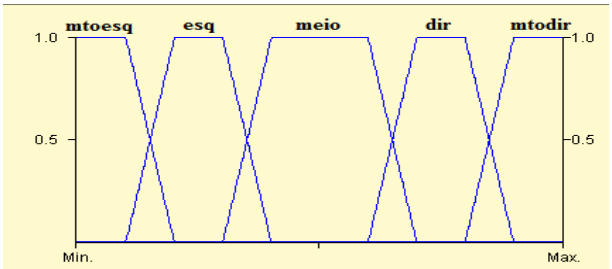

Figura 23. Conjunto Fuzzy para o eixo Y do goleiro.

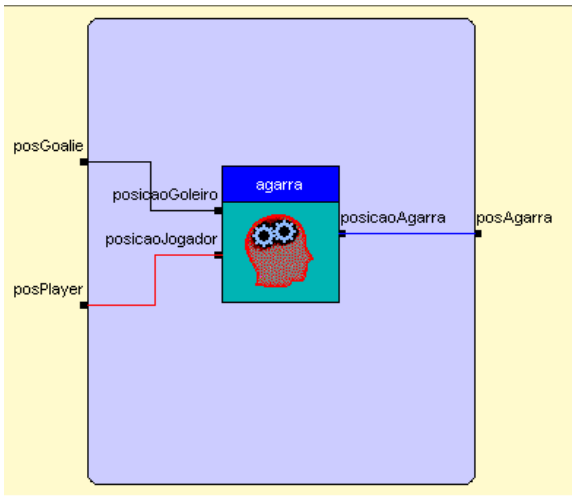

Figura 24. Máquina de Inferência do Primeiro Controlador.

4.4 Segundo Controlador Construído

Para tentar solucionar o problema do goleiro no eixo X, criou-se um controlador fuzzy específico para isso. Foi um controlador bem trivial: a medida que o atacante se aproximava, e o goleiro não pudesse agarrar a bola, ele iria recuando para cobrir melhor o gol.

As variáveis linguísticas para o atacante e para o goleiro foram: **mtoperto, perto, medio, longe** e **mtolonge** - figura 25. A máquina de inferência do controlador está representada na figura 26.

O controlador para o eixo Y não foi alterado; esperava-se que esse novo comportamento de recuar e adiantar fosse o suficiente. O goleiro continuava a tomar gols pelos cantos, ele recuava mas não ia suficientemente para as pontas.

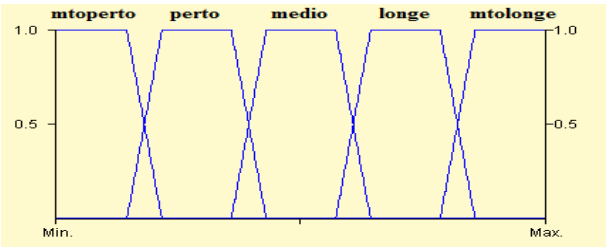

Figura 25. Conjunto Fuzzy para o eixo X do atacante e goleiro.

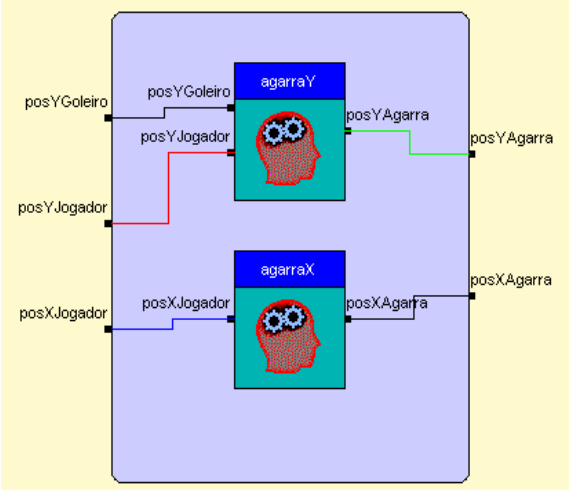

Figura 26. Máquina de Inferência do Segundo Controlador.

4.5 Terceiro Controlador Construído

Pensando-se em tornar o controlador mais eficiente, resolveu-se integrar o controlador do eixo Y com a entrada do controlador do eixo X e subdividir mais as variáveis linguísticas do jogador em relação ao eixo Y.

O controlador do eixo Y agora recebia como entrada a posição X do atacante também. Ele decidia a partir das três variáveis de entrada: **posYJogador**, **posXJogador** e **posYGoleiro**. Com isso, tentou-se evitar situações de risco, como quando o atacante estivesse muito a esquerda e muito perto; nesse caso, o goleiro deveria ir para muito a esquerda também para fechar o ângulo.

A lógica da proposta começou a se transformar, ao invés de simplesmente prever, o goleiro passou a tentar evitar o chute, fechando os ângulos.

A idéia inicial de que se o atacante viesse pela direita e o goleiro estivesse muito a direita, sendo o mais provável o atacante chutar a esquerda, e o goleiro se movimentar para tal posição, não funcionou; pois ao movimentar-se para esquerda nessa situação, o goleiro na

verdade abria o ângulo para o atacante chutar. Talvez, se esse posicionamento ocorresse no mesmo ciclo em que o atacante chutasse, essa lógica pudesse funcionar, mas não era o caso.

As variáveis linguísticas do atacante passaram a ser **mtoesq, esq, meio, dir** e **mtodir** para o eixo Y - figura 27 - e mantiveram-se inalteradas para o eixo X. Na figura 28 pode-se ver melhor as variáveis de entrada e saída da máquina de inferência.

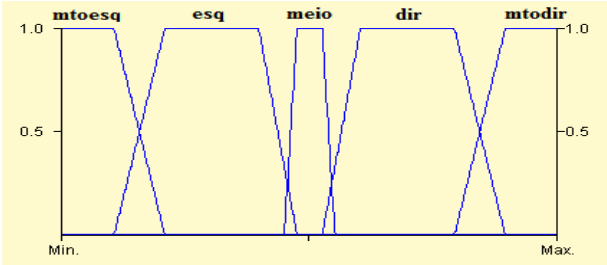

Figura 27. Conjunto Fuzzy para o eixo Y do atacante.

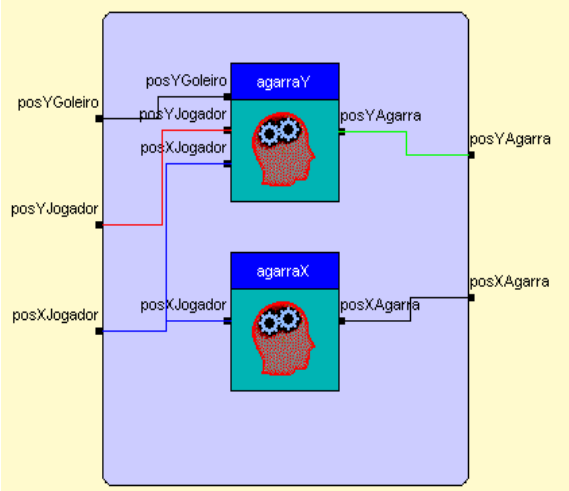

Figura 28. Máquina de Inferência do Terceiro Controlador.

4.6 Quarto Controlador Construído

Seguindo a nova lógica adotada do goleiro cobrir o ângulo para evitar o chute, o controlador foi totalmente remodelado, principalmente para o eixo Y.

O controlador do eixo Y passou a ter apenas uma variável de entrada e uma de saída. O valores das variáves linguísticas foram adaptados a esse novo enfoque, sendo mais refinados. A experiência com a construção dos controladores anteriores e a obsevação de seus comportamentos em inúmeras partidas possibilitaram a construção deste novo controlador de maneira mais criteriosa e com bases mais firmes. Ele recebe como entrada apenas a posição Y do atacante com a posse de bola e devolve a posição Y que o goleiro deverá assumir. A medida que o atacante se desloca no eixo Y, o goleiro vai acompanhando a movimentação e atualizando sua posição, tentando sempre estar entre o atacante com a bola e o gol.

No controlador do eixo X, modificou-se apenas os valores das variáveis linguísticas e as regras de produção da máquina de inferência. O goleiro passou a ficar mais recuado quando o atacante está próximo ao gol. Esse comportamento está presente na maioria dos times analisados, como o Brainstormers, ATH, Helios e Opu_Hana. Esses times deixam o goleiro praticamente na linha do gol.

As variáveis linguísticas de entrada e saída para ambos os eixos, a máquina de inferência e as regras de produção estão ilustradas nas figuras 29, 30, 31, 32, 33, 34 e 35, respectivamente.

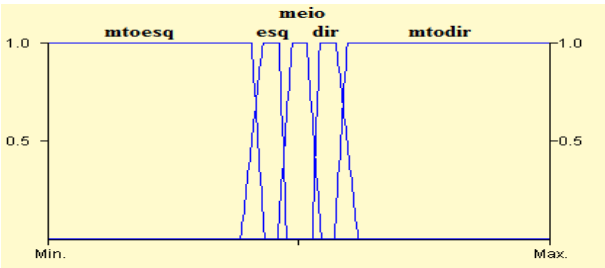

Figura 29. Conjunto Fuzzy para o eixo Y do atacante.

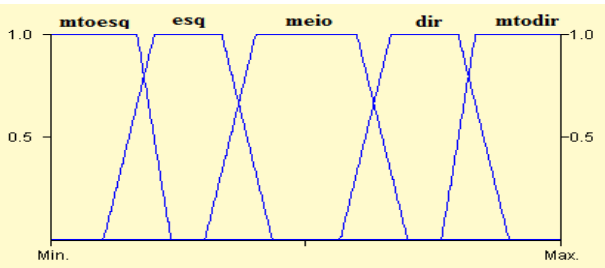

Figura 30. Conjunto Fuzzy para o eixo Y do goleiro.

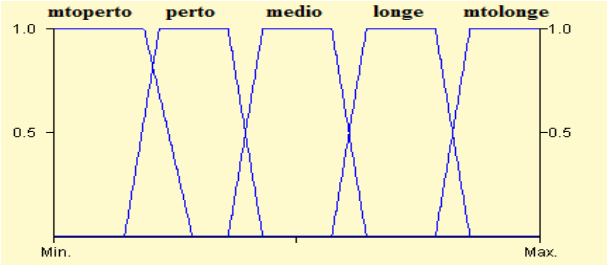

Figura 31. Conjunto Fuzzy para o eixo X do atacante.

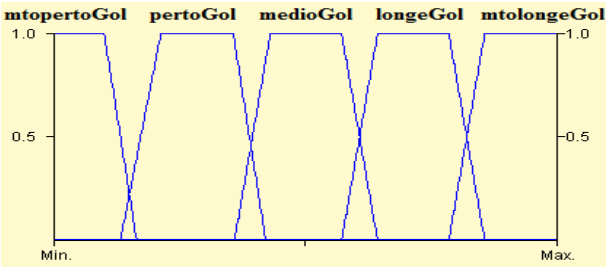

Figura 32. Conjunto Fuzzy para o eixo X do goleiro.

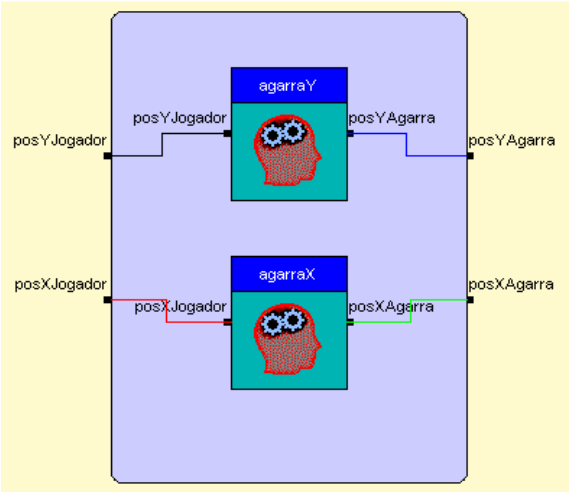

Figura 33. Máquina de Inferência do Quarto Controlador.

Rule			Premise		Conclusion
0	1.0	if	(posYJogador == mtoesquerda)	->	posYAgarra = mtoesquerda
1	1.0	if	(posYJogador == esquerda)	->	posYAgarra = esquerda
2	1.0	if	(posYJogador == centro)	->	posYAgarra = centro
3	1.0	if	(posYJogador == direita)	->	posYAgarra = direita
4	1.0	if	(posYJogador == mtodireita)	->	posYAgarra = mtodireita
*					

Figura 34. Regras para o Controlador do eixo Y.

Rule			Premise		Conclusion
0	1.0	if	(posXJogador == mtolonge)	->	posXAgarra = mtolongeGol
1	1.0	if	(posXJogador == longe)	->	posXAgarra = longeGol
2	1.0	if	(posXJogador == medio)	->	posXAgarra = medioGol
3	1.0	if	(posXJogador == perto)	->	posXAgarra = pertoGol
4	1.0	if	(posXJogador == mtoperto)	->	posXAgarra = mtopertoGol
*					

Figura 35. Regras para o Controlador do eixo X.

4.7 Resultados Obtidos

Foram simuladas vinte partidas contra o time que serviu de base para a criação do Bahia2D, o UvA_Base; contra o atual campeão brasileiro, o PET_Soccer; e contra quatro dos melhores times do mundo, o Brainstormers - campeão mundial, o ATH - sexto colocado, o Helios - terceiro colocado, e o Opu_Hana - quarto colocado.

Dez dessas partidas contra cada time foram utilizando a versão do Bahia2D que jogou na RoboCup Internacional 2007, realizada em Atlanta, nos Estados Unidos. Essa versão foi escolhida pois era a mais estável na época em que os testes começaram a ser realizados. Os resultados dessas partidas servem como base de comparação para avaliar o desempenho do controlador, eles fazem parte do grupo de controle.

As outras dez partidas contra cada time foram simuladas utilizando a versão do Bahia2D de Atlanta, acrescidas apenas do controlador fuzzy e com algumas alterações no código, já descritas nesta monografia.

Todas as partidas foram simuladas na mesma máquina - um Sony Vaio VGN-FZ160E, Intel Core 2 Duo T7300, 2Gb de RAM, 200Gb HD - para evitar possíveis diferenças devido a configurações distintas. Um processador lento, ou pouca memória, pode vir a acarretar na perda de ciclos, prejudicando o desempenho dos times

Uma vez de posse dos logs das partidas, contabilizou-se os gols sofridos e as

bolas agarradas pelo goleiro. A soma desses resultados representa a quantidade de chutes a gol realizadas pelo time adversário.

Calculou-se também o percentual de bolas agarradas por chutes. Entretanto, a eficiência do mesmo não deve ser medida apenas por esse percentual, visto que este pode ter diminuído e mesmo assim o número de gols levados pelo goleiro decrescido. Isso ocorre porque, contra alguns times, o goleiro acaba evitando vários chutes do atacante, tendendo a diminuir a quantidades de bolas chutadas a gol e, consequentemente, das próprias defesas e gols. Assim, um percentual menor não significa que o desempenho do goleiro piorou; todavia, ele deve ser levado em consideração como um parâmetro para medir seu desempenho.

Na tabela 2 pode-se ver a média de gols sofridos, bolas agarradas e chutes a gol contra cada um dos times testados e no total. Além do percentual de bolas agarradas por chutes a gol. Avaliando-se o conjunto desses parâmetros, tem-se condições de chegar a uma conclusão mais precisa quanto ao desempenho do goleiro e do controlador difuso utilizado. As tabelas com os resultados mais detalhados contra cada time encontram-se no Apêndice A.

Tabela 2. Resultados Totais.

	Sem Controlador				Com Controlador			
	Gols	Defesas	Chutes	Def./Chutes	Gols	Defesas	Chutes	Def./Chutes
ATH	82	108	190	56,84%	45	68	113	60,18%
Brainstormers	177	194	371	52,29%	106	205	311	65,92%
Helios	222	146	368	39,67%	285	151	436	34,63%
Opu_Hana	154	339	493	68,76%	192	327	519	63,01%
PET_Soccer	42	98	140	70,00%	54	133	187	71,12%
UvA_Base	25	308	333	92,49%	31	385	416	92,55%
TOTAL	702	1193	1895	62,96%	713	1269	1982	64,03%

Pode-se observar que contra o ATH, o percentual de defesas por chutes aumentou em quase 4%, o que já é considerado um resultado significativo, não sendo mais simplesmente uma diferença estatística. Entretanto, o principal indicativo da eficiência do controlador contra este time foi a quantidade de gols. A média sem o controlador de 8,2 gols por partida diminuiu para 4,5 gols, isso significa uma redução de 45,1%. Ou seja, utilizando o

controlador fuzzy o time deixou de tomar uma média de 3,7 gols por partida, o que muitas vezes é decisivo em uma competição.

Contra o Brainstormers o percentual de defesas aumentou ainda mais: foi de 52,29% para 65,92%, representando um aumento de 13,63%. Isso significa que o goleiro agarrou muito mais as bolas chutadas, indicando que o posicionamento dele funcionou muito bem contra o atual campeão mundial. A quantidade de gols também diminuiu consideravelmente, foi de 17,7 para 10,6 gols por partida – um ótimo resultado de 40,1%. Na figura 36, pode-se ver o bom posicionamento do goleiro em relação ao atacante, defendendo o gol e agarrando a bola; ele recua a medida que o atacante avança, fechando o ângulo do chute a gol, até que o atacante arrisca um chute que é defendido pelo goleiro.

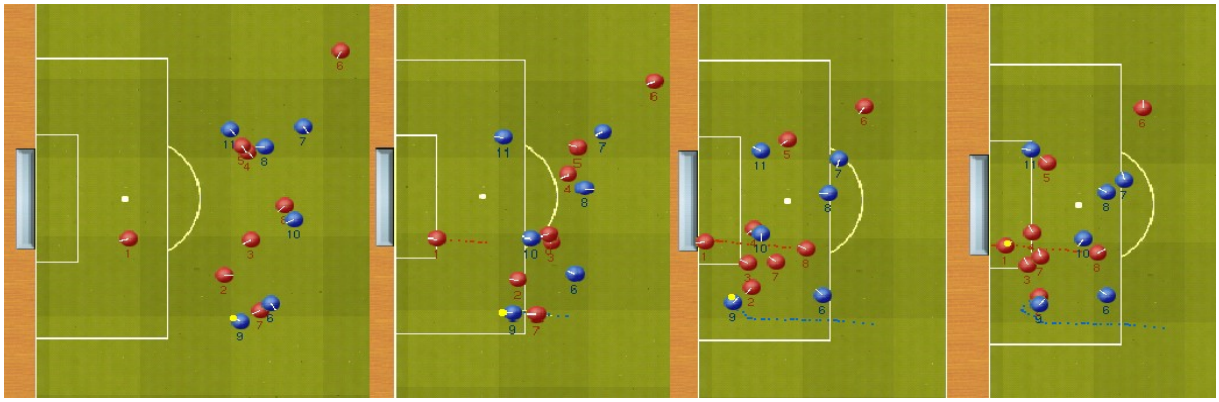

Figura 36. Goleiro defendendo o gol e agarrando a bola - contra o Brainstormers.

Contra ambos, o ATH e o Brainstormers, a quantidade de chutes a gol diminuiu, indicando que o goleiro realmente evitou diversos chutes a gol. O atacante, por não ter ângulo para chutar, acaba perdendo a bola: ou chuta errado ou para fora, ou dá a oportunidade para o zagueiro roubar a bola por ficar muito tempo indeciso. Essa situação é ilustrada na figura 37: o atacante se aproxima com a bola – trajetória azul - e o goleiro recua – trajetória vermelha - sempre fechando o ângulo, de forma que possibilita a maior aproximação dos zagueiros, fazendo com que o atacante acabe chutando para fora.

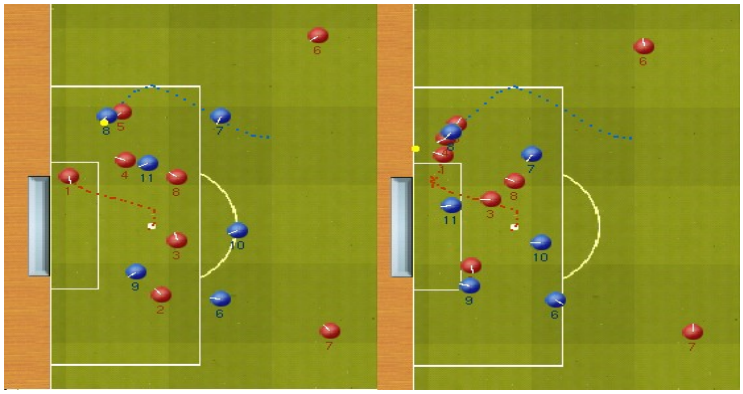

Figura 37. Goleiro impedindo chute a gol - contra o Brainstormers.

Contra o Helios e o Opu_Hana o percentual diminuiu um pouco, em torno de 5%, e os gols sofridos aumentaram; indicando que o controlador não foi muito eficiente contra esses times de maneira geral. Isso aconteceu por tais times terem duas característica muito fortes: alta velocidade e tocar muito a bola no ataque. Um atacante deles com a posse de bola raramente chuta direto a gol; na maioria das vezes, eles tocam muito rapidamente entre si antes de algum deles chutar a gol, tornando a situação bastante complicada para o goleiro defender sozinho, chegando a ser, muitas vezes, até impossível. Nesses casos, o goleiro deveria contar com o apoio de sua zaga: ela deveria fazer a marcação dos adversários e se posicionar entre os demais atacantes e o gol, o que, infelizmente, ocorre muito pouco no Bahia2D ainda, acarretando neste pobre desempenho do goleiro com o controlador. Na figura 38, pode-se observar o posicionamento correto do goleiro em relação ao atacante com posse de bola e os companheiros deste atacante prontos para receber a bola e chutar a gol, anulando o goleiro.

Figura 38. Bom posicionamento do goleiro, mas falta de marcação – contra o Helios.

Contra o PET_Soccer e o UvA_Base o percentual permaneceu praticamente inalterado, aumentando levemente nos dois casos. A quantidade de gols também aumentou um pouco. Esses dois times encontram-se em um nível mais a baixo dos demais testados, eles pouco ou nada analisam antes de chutar a gol; se estiverem perto, quase que independentemente das condições, eles chutam. Muitas vezes o PET_Soccer chega a tocar a bola antes de chutar a gol, mas o UvA_Base nem consegue a fazer isso. Assim, já era esperado que o controlador não diminuísse a quantidade de chutes a gol contra eles. Então, como o percentual de bolas agarradas não diminuíu, o resultado é considerado satisfatório.

Um fator relevante para a análise dos resultados é a deficiência apresentada pelo time Bahia2D na reposição de bola do goleiro. O trabalho de Pereira (2007) foi justamente criar controladores fuzzy para auxiliar na escolha de reposição de bola: uma vez com a posse de bola, o goleiro escolhe entre 3 cones com menos oponentes e então entre os companheiros menos marcados para repor a bola. Mas nem sempre esta solução funciona, algumas vezes o goleiro repõe a bola diretamente no pé do atacante adversário, causando um gol quase que inevitável. Quanto mais vezes ele repõe a bola, maior a quantidade possível de erros. Assim, contra times que chutam muito e que o goleiro agarra muito, as chances de um erro na reposição são enormes. Contra o UvA_Base ocorre pelo menos um erro como este em quase todas as partidas, o que prejudica os resultados do controlador, pois quanto mais o controlador funciona e o goleiro agarra a bola, mais gols de erro de reposição o time deve levar. Contra o PET_Soccer ocorre o mesmo. Isso explica porque a quantidade de gols aumentou um pouco e ainda asssim o percentual de bolas agarradas também aumentou, mantendo o resultado quase inalterado contra este dois times. Com o controlador nessas condições, por um lado o goleiro agarra mais bolas, por outro ele erra mais sua reposição. Essa contrapartida ameniza a diferença de resultados, fazendo que o rendimento do controlador aparente ser mais baixo do que realmente é. Consertendo-se este problema de reposição, melhorar-se-á tanto o time sem o controlador, quanto mais ainda o time com o controlador.

Analisando o resultado total contra os seis times, o controlador não apresentou

nem uma melhora, nem um piora significativa. O resultado manteve-se praticamente o mesmo, aumentando o percentual de bolas agarradas em 1%. Com isso, conclui-se que o controlador não piorou o desempenho do time e, sabendo-se utilizá-lo, ele pode provocar significativas melhorias. É preciso saber em que partidas ativá-lo. O próprio goleiro deve aprender a utilizar ou não o controlador, a depender do adversário e do momento. Havendo essa escolha criteriosa, o time tem muito a ganhar com o uso do controlador apresentado neste trabalho, como comprovado contra o Brainstormers e o ATH. Se o goleiro soubesse decidir em que situações não utilizar o controlador contra o Helios e o Opu_Hana os resultados não teriam sido prejudicados por ele e possivelmente seu rendimento melhoraria. Mais detalhes sobre essas e outras possibilidades para aumentar o desempenho do goleiro, utilizando este mesmo controlador, serão mostrados nos próximo capítulo sobre as considerações finais e os trabalhos futuros.

5 CONSIDERAÇÕES FINAIS

Esta monografia apresentou os resultados da pesquisa sobre controladores difusos para o posicionamento do goleiro em relação ao atacante, no time Bahia2D.

Com a evolução dos controladores fuzzy construídos, chegou-se a uma versão estável, que gerou uma melhora no posicionamento do goleiro, aumentando seu rendimento em determinados casos.

Contra o ATH e o Brainstormers os resultados foram mais significativos, indicando uma boa performance. Deve-se estudar mais profundamente as características destes times para assimilar exatamente as situações em que o controlador deve ser usado.

Contra o Helios e o Opu_Hana, times mais ofensivos, os resultados diminuíram um pouco, indicando situações em que o controlador não deve ser usado ou, se usado, contar com o apoio da zaga. O goleiro passou a fechar melhor o ângulo de chute a gol do atacante, mas, ao agir assim, acaba deixando outras partes do gol mais desprotegidas, delegando esta tarefa à zaga, que não a cumpre corretamente; estes times, ao tocarem muito antes de chutar, exploram esta falha.

Contra o PET_Soccer e o UvA_Base não houve uma diferença muito significativa, aumentando um pouco o percentual de bolas agarradas pelo goleiro, mas também ligeiramente o número de gols sofridos.

Sugere-se como trabalhos futuros, um estudo onde o próprio goleiro aprenda quando utilizar o controlador para tirar o melhor proveito do que ele tem para oferecer. Isso pode ser alcançado utilizando-se redes neurais ou aprendizado por reforço. Tal tema já faz parte dos objetivos do grupo do Bahia Robotics Team, e a autora pretende continuar com esta pesquisa pessoalmente.

Uma outra idéia para melhorar o desempenho do controlador, é inserir um novo controlador para o posicionamento do goleiro em relação à bola. Assim, com ambos

controladores agindo em conjunto, as falhas do goleiro serão melhor abrangidas e a prababilidade de sucesso deverá ser maior. Através de técnicas de aprendizado de máquina, que poderia ser redes neurais e aprendizado por reforço também, o goleiro poderia decidir quando deveria usar cada um dos controladores a depender da situação. Um outro estudo que faz parte dos objetivos do BRT.

Para otimizar o controlador já construído, soluções através de algorítimos genéticos já estão sendo estudadas. Objetiva-se melhorar o valor das funções de pertinência, tornado o controlador mais preciso. Diversos valores das funções de pertinência, para um ou mais tipos definidos, formariam a população. Esta população cruzaria, mutaria e migraria, melhorando o *fitness* cada vez mais. O *fitness* seria o percentual de bolas agarradas por chutes a gol. Esta pesquisa inclusive já vem sendo desenvolvida pela autora, devendo apresentar resultados, ainda que parciais, até o final do mês em curso.

A distância para a bola ser considerada sob posse de um jogador também poderia ser otimizada através de redes neurais ou bayesianas. A distância para que o goleiro passe a correr sempre atrás da bola também se encaixa neste caso, como outros valores absolutos que foram estipulados empiricamente neste trabalho monográfico. A idéia é achar os valores ótimos para estes casos.

Uma outra melhoria importante é o trabalho em conjunto do goleiro com a zaga. Os zagueiros devem auxiliar o goleiro, fechando o ângulo dos outros atacantes possíveies de receber a bola para chutar a gol, além de os marcar intensivamente. Existem situações que o goleiro sozinho nada pode fazer contra um grupo de atacantes, sendo imprescindível essa colaboração da defesa de seu time. Adicionar um nível cognitivo seria o ideal, para que eles pudessem se comunicar e combinar a marcação do ataque de forma coletiva. O trabalho de Cruz (2007) implementou a marcação pelos zagueiros, faltando apenas o terceiro tipo de marcação, que foca justamente no problema ora indicado. O time Bahia2D atualmente apresenta este lapso, o que acabou por prejudicar os resultados encontrados nesta monografia para o controlador difuso construído. Tal pesquisa, bem como a correção da reposição de

bola, fazem parte do escopo do BRT e serão tratadas até a RoboCup 2008, a ser realizada na China.

Assim, pode-se perceber que esta pesquisa serve como base para inúmeros trabalhos futuros, sendo de grande relevância para o time Bahia2D pois comprovou que pode-se otimizar o comportamento sem bola do goleiro através de controladores fuzzy, esperando-se que venha a contribuir também para o desenvolvimento da robótica inteligente em geral.

REFERÊNCIAS

BARONE, Dante. **Sociedades Artificiais: A Nova Fronteira da Inteligência nas Máquinas.** Brasil: Bookman, 2003.

BARRETO, J. M.. **Redes Neurais: Fundamentos e Aplicações**. In: Minicurso do II Simpósio Brasileiro de Automação Inteligente. Curitiba: CEFET, 1995.

BARRETO, J. M.. **Introdução às Redes Neurais Artificiais**. In: V Escola Regional de Informática. SBC, Regional Sul. Brasil, 1997.

BITTENCOURT, Guilherme. **Inteligência Artificial: Ferramentas e Teorias.** Brasil: Editora da UFSC, 1998.

CHUA, W. F.. **Radical Developments in Accounting Thought**. The Accounting Review, vol. 61, n° 4., p. 601-632, 1986.

COHEN, P. R.; LEVESQUE, H. J.. **Intention = Choice + Commitment**. Proceedings of AAAI 87, Seattle, 1987. 410-415 p.

CRUZ, Danilo. **Controladores Nebulosos e Rede Neural para a Marcação de um Agente Jogador de Futebol do Ambiente Simulado 2D.** Trabalho de conclusão de curso entregue à Universidade do Estado da Bahia. Salvador, 2007.

DURFEE, E.H.. **The Distributed Artificial Intelligence Melting Pot**. IEEE Transactions on Systems, Man and Cybernetics, 21(6): 1301-1306 p., 1991. Special Issue on Distributed Artificial Intelligence.

ENGELBRECHT, Andries P.. **Fundamentals of Computational Swarm Intelligence.** Inglaterra: Wiley, 2005.

GLASER, B.; STRAUSS, A. L.. **The Discovery of Grounded Theory**. Chicago: Aldine de Gruyter, 1967.

GOLLIN, Michael. **Implementation einer Bibliothek für Reinforcement Learning und Anwendung in der RoboCup Simulationsliga**. Trabalho de conclusão de curso entregue à Humbolt-Universität zu Berlin. Berlin, 2005.

GONÇALVES, Eder; LOUREIRO, Augusto; BITTENCOURT, Guilherme. **Otimização de Controladores Nebulosos e Sistemas Especialistas Reativos Utilizando Algoritmos Genéticos**. Anais do XXI Congresso da SBC - ENIA - Encontro Nacional de Inteligência Artificial. Brasil, 2001.

KASABOV, Nikola K. **Foundations of Neural Networks, Fuzzy Systems, and Knowledge Engeneering**. Massachusetts Institute of Technology. Londres, 1998.

KOK, Jelle R.; VLASSIS, Nikos; GROEN, Frans. **UvA Trilearn 2003 Team Description**. RoboCup 2003 Team Description Papers CD-ROM, 2003.

KOK, Jelle R; VLASSIS, Nikos. **UvA Trilearn 2005 Team Description.** RoboCup 2005 Team Description Papers CD-ROM, 2005.

KURZWEIL, Ray. **A Era das Máquinas Espirituais.** Brasil: Aleph, 2007.

LIMA, P.; Custodio, L.; AKIN, L.; Jacoff, A.; KRAERTZSCHMAR, G.; KIAT, N. G.; OBST, O.; RÖFFER, T.; TAKAHASHI, Y.; ZHOU, C.. **Robocup 2004 Competitions and Symposium: a Small Kick for Robots, a Giant Score for Science**. AI Magazine, Summer 2005, 36-61 p.. AAAI.

LOUREIRO, Augusto. **O Futebol de Robôs: uma Ferramenta para Pesquisa e o Ensino em Automação e Informática Industrial**. Minicurso apresentado na Semana Universitária da UNIFACS. Salvador, 2002.

NAKASHIMA, T.; TAKATANI, M.; UDO, M.; ISHIBUSHI. **H. Hana Team Description**. In: RoboCup 2004: Robot Soccer World Cup VIII. Lecture Notes in Computer Science 3276, Team Description Papers CD-ROM. Berlin : Springer-Verlag, 2004.

NEWELL, A.. **Physical Symbol Systems**. Cognitive Science, 4: 135-183 p, 1980.

PEREIRA, Jailsa Oliveira. **Desenvolvimento de Controladores Difusos para o Goleiro do Futebol de Robôs Simulado**. Trabalho de conclusão de curso entregue à Universidade do Estado da Bahia. Salvador, 2007.

PIANESSO, Ana; ROCHA, João Paulo; LINDER, Marcelo; BITENCOURT, Andréia; JÚNIOR, Julhilson; PERRONE, Ricardo; LOUREIRO, Augusto. **Oxente Team: Implementação de Habilidades de Jogo em um Time de Futebol de Robôs Simulado Utilizando Otimização Heurística**. II IEEE Latin America Robotic Symposium/RoboCup Latin America: VI SBAI. Brasil, 2005.

POST, E.. **Formal Reductions of the General Combinatorial Problem**. American Journal of Mathematics, 65: 197-268 p., 1943.

RABUSKE, Renato Antônio. **Inteligência Artificial.** Brasil: Editora da UFSC, 1995.

REZENDE, Solange Oliveira. **Sistemas Inteligentes: Fundamentos e Aplicações.** Brasil: Manole, 2005.

RIEDMILLER, M.; GABEL, T.; KNABE, J.; STRASDAT H.. **Brainstormers 2D – Team Description 2005**. RoboCup 2005 Team Description Papers CD-ROM, 2005.

RIEDMILLER, M.; MERKE, A.; WITHOPF, D.. **Brainstormers 2004 Team Description**. RoboCup 2004: Robot Soccer World Cup VIII. Lecture Notes in Computer Science 3276, Team Description Papers CD-ROM. Berlin : Springer-Verlag, 2004.

RUSSEL, Stuart; NORVING, Peter. **Atificial Intelligence: A Modern Approach**. 2. ed, Nova Deli: Prentice-Hall India, 2006.

SANDRI, Sandra; CORREA, Cláudio. **Lógica Nebulosa.** V Escola de Redes Neurais, ITA. São José dos Campos, 1999.

SICHMAN, J. S.. **A Model for the Decision Phase of Autonomous Belief Revision in Open Multi-agent System**. Journal of Brazilian Computer Soceity, 3(1): 40-50 p.. 1996. ISSN 0104-6500.

SILVA, Hugo; MEYER, Jessica; SOUZA, Victor; VIEGAS, Simon; SIMÕES, Marco; ARAGÃO, Helder. **Controladores Fuzzy Para Agentes Robôs Jogadores De Futebol**. In: Workshop de Trabalhos de Iniciação Científica e Graduação Bahia-Alagoas-Sergipe. Anais da 7a Escola Regional de Computação Bahia-Alagoas-Sergipe. Porto Alegre: SBC, 2007.

SILVA, Hugo; SIMÕES, Marco; ARAGÃO, Helder. **Desenvolvimento de Controladores Fuzzy Para Robôs De Futebol Simulado Do Tipo Atacante**. In: Workshop de Trabalhos de Iniciação Científica e Graduação Bahia-Alagoas-Sergipe. Anais da 7a Escola Regional de Computação Bahia-Alagoas-Sergipe. Porto Alegre : SBC, 2007.

STANKEVICH, L.; KRITCHOUN, A.; IVANOV, A.; SEREBRYAKHOV, S.. **Zenit-NewERA Team Description**. In: RoboCup 2004: Robot Soccer World Cup VIII. Lecture Notes in Computer Sciense 3276, Team Description Papers CDROM. Berlin : Springer-Verlag, 2004.

STONE, Peter; RILEY, Patrick; VELOSO, Manuela. **The CMUnited-99 Champion Simulator Team**. 1999.

STRAUSS, A. L.; CORBIN, J. M.. **Basics of Qualitative Research: Grounded Theory Procedures and Techniques**. Sage Publications, Newbury Park, 1990.

VIEIRA, Orivaldo; ROCHA, João Paulo; LINDER, Marcelo; LOUREIRO, Augusto. **MecaTeam: Um Sistema Multiagente Para O Futebol De Robôs Simulado Baseado No Agente Autônomo Concorrente.** Anais do XXVI Congresso da SBC – EnRI – III Encontro de Robótica Inteligente. Brasil, 2006. 146-152 p..

VIEIRA, Orivaldo; LOUREIRO, Augusto. **MecaTeam 2006: Um Sistema Multiagente Reativo para o Futebol de Robôs Simulado**. In: Workshop de Trabalhos de Iniciação Científica e Graduação Bahia-Alagoas-Sergipe. Anais da 7a Escola Regional de Computação Bahia-Alagoas-Sergipe. Porto Alegre : SBC, 2007.

XAVIER, Raphael O.; BARBOSA, Rodrigo; MATSURA, Jackson P.. **O Time De Futebol Simulado ITANDROIS-2D**. Anais do XXVI Congresso da SBC – EnRI – III Encontro de Robótica Inteligente. Brasil, 2006. 153-159 p..

APÊNDICE A – Tabelas com análise dos jogos contra cada time

ATH								
	Sem Controlador				Com Controlador			
	Gols	Defesas	Chutes	Def./Chutes	Gols	Defesas	Chutes	Def./Chutes
1a Partida	7	10	17	58,8%	7	10	17	58,8%
2a Partida	9	14	23	60,9%	4	5	9	55,6%
3a Partida	12	5	17	29,4%	7	12	19	63,2%
4a Partida	6	13	19	68,4%	4	6	10	60,0%
5a Partida	12	10	22	45,5%	4	2	6	33,3%
6a Partida	6	14	20	70,0%	4	8	12	66,7%
7a Partida	11	11	22	50,0%	3	10	13	76,9%
8a Partida	6	16	22	72,7%	5	3	8	37,5%
9a Partida	6	11	17	64,7%	4	11	15	73,3%
10a Partida	7	4	11	36,4%	3	1	4	25,0%
TOTAL	82	108	190	56,84%	45	68	113	60,18%

Brainstormers								
	Sem Controlador				Com Controlador			
	Gols	Defesas	Chutes	Def./Chutes	Gols	Defesas	Chutes	Def./Chutes
1a Partida	17	16	33	48,5%	9	23	32	71,9%
2a Partida	17	15	32	46,9%	14	20	32	71,9%
3a Partida	27	19	46	41,3%	12	22	34	58,8%
4a Partida	21	26	47	55,3%	10	15	34	64,7%
5a Partida	15	21	36	58,3%	8	17	25	60,0%
6a Partida	14	27	41	65,9%	9	22	25	68,0%
7a Partida	13	27	40	67,5%	14	23	31	71,0%
8a Partida	15	10	25	40,0%	11	32	37	62,2%
9a Partida	16	19	35	54,3%	11	17	43	74,4%
10a Partida	22	14	36	38,9%	8	14	28	60,7%
TOTAL	177	194	371	52,29%	106	205	311	65,92%

Helios								
	Sem Controlador				Com Controlador			
	Gols	Defesas	Chutes	Def./Chutes	Gols	Defesas	Chutes	Def./Chutes
1a Partida	25	13	38	34,2%	17	15	32	46,9%
2a Partida	17	13	30	43,3%	29	13	42	31,0%
3a Partida	22	13	35	37,1%	25	16	41	39,0%
4a Partida	17	17	34	50,0%	36	19	55	34,5%
5a Partida	26	14	40	35,0%	32	19	51	37,3%
6a Partida	20	16	36	44,4%	25	13	38	34,2%
7a Partida	23	17	40	42,5%	33	14	47	29,8%
8a Partida	24	10	34	29,4%	32	13	45	28,9%
9a Partida	22	16	38	42,1%	22	15	37	40,5%
10a Partida	26	17	43	39,5%	34	14	48	29,2%
TOTAL	222	146	368	39,67%	285	151	436	34,63%

Opu_Hana								
	Sem Controlador				Com Controlador			
	Gols	Defesas	Chutes	Def./Chutes	Gols	Defesas	Chutes	Def./Chutes
1a Partida	20	35	55	63,6%	18	37	55	67,3%
2a Partida	12	35	47	74,5%	21	26	47	55,3%
3a Partida	17	31	48	64,6%	22	27	49	55,1%
4a Partida	13	43	56	76,8%	21	37	58	63,8%
5a Partida	14	31	45	68,9%	17	36	53	67,9%
6a Partida	12	36	48	75,0%	24	26	50	52,0%
7a Partida	14	32	46	69,6%	18	31	49	63,3%
8a Partida	14	36	50	72,0%	21	33	54	61,1%
9a Partida	14	29	43	67,4%	15	46	61	75,4%
10a Partida	24	31	55	56,4%	15	28	43	65,1%
TOTAL	154	339	493	68,76%	192	327	519	63,01%

PET_Soccer								
	Sem Controlador				Com Controlador			
	Gols	Defesas	Chutes	Def./Chutes	Gols	Defesas	Chutes	Def./Chutes
1a Partida	7	12	19	63,2%	8	10	18	55,6%
2a Partida	2	10	12	83,3%	4	17	21	81,0%
3a Partida	4	9	13	69,2%	7	14	21	66,7%
4a Partida	8	4	12	33,3%	6	6	12	50,0%
5a Partida	2	4	6	66,7%	7	21	28	75,0%
6a Partida	2	10	12	83,3%	8	12	20	60,0%
7a Partida	4	11	15	73,3%	1	10	11	90,9%
8a Partida	2	13	15	86,7%	1	10	11	90,9%
9a Partida	4	10	14	71,4%	6	13	19	68,4%
10a Partida	7	15	22	68,2%	6	20	26	76,9%
TOTAL	42	98	140	70,00%	54	133	187	71,12%

UvA_Base								
	Sem Controlador				Com Controlador			
	Gols	Defesas	Chutes	Def./Chutes	Gols	Defesas	Chutes	Def./Chutes
1a Partida	4	34	38	89,5%	1	32	33	97,0%
2a Partida	4	36	40	90,0%	7	62	69	89,9%
3a Partida	1	32	33	97,0%	2	46	48	95,8%
4a Partida	0	33	33	100,0%	3	22	25	88,0%
5a Partida	2	24	26	92,3%	1	39	40	97,5%
6a Partida	2	27	29	93,1%	3	45	48	93,8%
7a Partida	2	32	34	94,1%	3	25	28	89,3%
8a Partida	2	32	34	94,1%	6	51	57	89,5%
9a Partida	5	24	29	82,8%	4	36	40	90,0%
10a Partida	3	34	37	91,9%	1	27	28	96,4%
TOTAL	25	308	333	92,49%	31	385	416	92,55%