Uma linguagem para definição de comportamento de robôs jogadores de futebol no ambiente TeamBots™

Edson Elmar Schlei (FURB/BCC)
tatu@tpa.com.br

Jomi Fred Hübner (FURB/DSC/GIA)
jomi@inf.furb.br

Resumo. Este trabalho apresenta o desenvolvimento de uma linguagem para a construção de times de robôs formada por agentes reativos. Mais especificamente, são apresentadas as características mais relevantes desta linguagem e seus processos de especificação, implementação e utilização pelos agentes. Como resultado, tem-se uma linguagem de descrição de comportamentos de agentes jogadores de futebol que funcionam em um simulador da RoboCup.

Palavras-chave: RoboCup, Ambiente TeamBots, times de robôs, agentes jogadores de futebol, arquiteturas de agentes reativos.

1 Introdução

A construção dos agentes que controlam os robôs jogadores de futebol em ambientes simulados é uma tarefa complexa e que envolve conhecimento em diversas áreas. O deslocamento do robô dentro do campo é a tarefa básica do agente. O agente deve controlar o robô, tendo o conhecimento do lado que é o gol adversário e onde estão posicionados os seus companheiros de equipe. A detecção da posição bola e dos oponentes é outra tarefa que o agente deve saber fazer para poder ir ao encontro da bola. Após o agente ter a bola em seu domínio, ele deve levá-la em direção do gol ou passar ela para outro agente que faça parte de sua equipe, como também driblar um agente do time adversário para alcançar o seu objetivo que é o gol adversário.

Num time de futebol não se pode só pensar no ataque (agentes atacantes) e em fazer gols. Tem-se que ter uma estrutura de defesa que deve impedir que o time adversário possa alcançar o seu objetivo. Sendo assim, tem-se que ter agentes que irão compor a defesa do time, na qual existe um agente denominado goleiro e outros irão compor a zaga.

Além dos papéis de ataque e defesa, também existe a estratégia do jogo. Tal estratégia também é esquematizada em função dos objetivos a alcançar, que é ganhar o jogo. Para a definição destas estratégias é preciso primeiramente definir o comportamento de cada um dos agentes que vai compor a equipe. Para descrever estes comportamentos, propõe-se a elaboração de uma linguagem de alto nível de abstração com as características necessárias para a definição do comportamento

dos agentes jogadores de futebol. Utilizando esta linguagem, o programador/estrategista pode definir a parte da estratégia que cabe a cada a agente-jogador. Com isso pode-se facilmente alterar o comportamento dos jogadores e consequentemente a estratégia do time. A facilidade decorre do fato de que mudanças podem ser feitas em uma linguagem de alto nível e propósito específico aqui proposta.

Considerando o objetivo do trabalho, a seção 2 apresenta a competição de futebol de robôs denominado de RoboCup e a categoria na qual este trabalho atua. Na seção 3 é apresentado o ambiente TeamBots™ utilizado no desenvolvimento dos agentes jogadores. A seção 4 apresenta a linguagem desenvolvida neste trabalho e, por fim, a seção 5 relata as conclusões do trabalho.

2 Futebol de Robôs

O xadrez foi um dos primeiros jogos onde foi aplicada a inteligência artificial. O desenvolvimento de máquinas que pudessem jogar sem o auxílio humano começou em meados dos anos 60. Russos e Americanos promoviam confrontos entre seus engenhos e grandes jogadores para saber qual das duas potências era mais eficiente na área computacional. No entanto, a máquina demorou a vencer o ser humano. Só em 1997, o super computador *Deep Blue*, da IBM, derrotou o campeão mundial Garry Kasparov. O computador usava inteligência artificial do tipo informação perfeita, ou seja, com um número limitado de possibilidades.

Dado que o problema de jogar xadrez foi quase resolvido, buscando um novo desafio, um grupo internacional de pesquisadores em Inteligência Artificial e Robótica propõe um novo problema a ser solucionado: uma partida de futebol de robôs (LCMI, 2000; RoboCup 2001). No futebol o número de fatores a ser analisado é quase infinito, até o atrito influi, o que torna o jogo mais complexo e impossibilita o uso das técnicas utilizadas pelos programas que jogam xadrez (LCMI, 1998; David, 2001). Desta iniciativa, assim como acontece no futebol jogado por humanos, surgiu a RoboCup "*Robo World Cup*", onde os jogadores são artificiais.

Este domínio de aplicação permite que diversas técnicas sejam testadas e, principalmente, comparadas. A construção de um time de futebol de robôs envolve a integração de diversas tecnologias, tais como: projeto de agentes autônomos, cooperação em sistemas multi-agentes, estratégias de aquisição de conhecimento, engenharia de sistemas de tempo real, sistemas distribuídos, reconhecimento de padrões, aprendizagem, controle de processos, etc. (LCMI, 2000).

A RoboCup possui três categorias: duas delas envolvem disputas entre times de robôs reais, pequenos (*small size league*) e médios (*middle size league*) e uma terceira envolve partidas disputadas em um simulador, disponível na Internet. Esta última categoria permite que grupos de pesquisadores em Inteligência Artificial desenvolvam times através da implementação de agentes computacionais autônomos capazes de cooperar para disputar uma partida de futebol de robôs, sem se preocupar com a parte física da construção de robôs (LCMI, 2000).

A primeira *"Robot World Cup"* aconteceu em agosto de 1997, em Nagoya, Japão, durante a *Fifteenth International Joint Conference on Artificial Intelligence* (IJCAI'97) e contou com a participação de pelo menos 40 times. Desde então as competições vêm acontecendo anualmente na IJCAI ou no *International Conference on Multi-Agent Systems* (ICMAS) com a participação de pesquisadores de todo o mundo (LCMI, 2000).

2.1 Categoria robôs de pequeno porte (F-180)

Na categoria de robôs de pequeno porte, com cinco robôs em cada time, as competições são disputadas em campo, 152,5 cm x 274 cm, verde. São permitidos: tanto um sistema de visão global com uma câmera no teto, como sistema distribuído de visão, onde cada um dos robôs tem sua própria câmera embarcada (LCMI, 2000)[1].

Os robôs não devem ultrapassar uma área de 180 cm^2. O robô deve caber dentro de um cilindro de 18 cm de diâmetro. Em se tratando de um robô cuja área da base assume o formato retangular a diagonal deve ser inferior a 18 cm. A máxima altura deve ser de 15 cm, se o time optar por um sistema de visão global, ou 22,5 cm se este utilizar a visão embarcada (LCMI, 2000).

A comunicação entre os robôs não sofre qualquer tipo de restrição, o que permite a utilização de estratégias de cooperação mais elaboradas. Nesta categoria, os desafios envolvidos englobam várias áreas da Automação Industrial, Inteligência Artificial, Robótica, Controle de Processos, Reconhecimento de Padrões, Sistemas de Tempo Real, Sistemas Distribuídos, etc. (LCMI, 2000).

3 TeamBots™

TeamBots é um conjunto de programas e pacotes Java para pesquisadores em robótica móvel na área de SMA. TeamBots é distribuído com o seu código fonte aberto. Atualmente os robôs desenvolvidos no TeamBots podem ser executados nos robôs que utilizam a tecnologia Nomadic, robô Nomad 150 (Balch, 2000).

Uma das mais importantes características do ambiente TeamBots é o suporte a prototipação e simulação do mesmo sistema de controle que é executado em robôs móveis. O ambiente TeamBots é bastante flexível suportanto a execução de múltiplos robôs heterogêneos com sistemas de controles heterogêneos. Ambientes experimentais complexos (ou simples) podem ser criados com paredes, estradas, outros robôs e obstáculos circulares. Todos esses objetos podem ser criados apenas editando-se um arquivo de configuração.

3.1 TBSim

TBSim faz parte do pacote de programas do ambiente TeamBots™. TBSim é um programa que tem por objetivo realizar a simulação das condições encontradas

[1] No capitulo sobre TeamBots será apresentado um programa simulador para esta categoria de robôs que é utilizado neste trabalho.

no mundo real (obstáculos, outros robôs, bola de golfe, tamanho da área de atuação, etc...) para robôs da categoria de médio porte utilizada nas competições da RoboCup. A simulação de tamanho e percepção é compatível com as especificações dos regulamentos da RoboCup para a categoria de robôs de pequeno porte. A Fig. 1 apresenta a tela do TBSim em execução.

Figura 1: Simulador TBSim

4 A linguagem

Levando em consideração os aspectos relacionados a futebol de robôs e o ambiente TeamBots abordados anteriormente, e sendo o objetivo deste trabalho o de desenvolver uma linguagem de descrição de vários tipos de comportamentos, seguem as propriedades mais relevantes que tal linguagem deve possuir:

a) o usuário tem que poder descrever o campo de futebol (tamanho do campo, área do gol, área de defesa, etc.);

b) verificar o estado do agente jogador (se está com a posse da bola, sua posição no campo, etc.);

c) usar ações primárias do agente jogador (andar, parar, girar, etc.);

d) descrever comportamentos para o agente jogador, sendo um comportamento uma composição coerente de ações primárias (comportamento de defesa, ataque, goleiro, etc.);

e) descrever rotinas que utilizem as ações primárias do agente e possam ser reutilizadas em vários comportamentos;

f) controlar a ativação de comportamentos diferentes para o mesmo agente, em outras palavras, qual comportamento deve ser ativado em função das condições do jogo.

4.1 Visão geral do uso da lingaguem

Com base na Fig. 2, que apresenta a visão geral do funcionamento da linguagem desenvolvida neste trabalho, está seção apresenta os passos que são necessários para a utilização da linguagem na criação de comportamentos para os agentes jogadores.

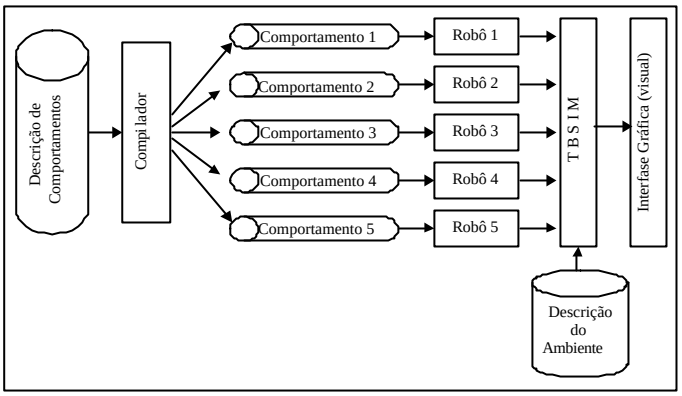

Figura 2: Visão geral da linguagem

Inicialmente é preciso criar um arquivo (representado pela caixa "Descrição de Comportamentos") no qual vai estar a descrição dos comportamentos dos cinco robôs. Este arquivo deve seguir a sintaxe definida para a linguagem (por razões de espaço a BNF da linguagem não é apresentada neste artigo, mas pode ser obtida em (Schlei, 2002)).

Após a descrição dos comportamentos dos agentes jogadores, é necessária a verificação dos comportamentos descritos. Para este propósito foi criado um compilador (representado pela caixa "Compilador"). Este compilador foi desenvolvido a partir do gerador de *parsers* JavaCC (novamente, por não ser o foco do artigo, maiores informações sobre o uso desta ferramenta podem ser obtidas em (Schlei, 2002)). Não encontrando erros no arquivo de comportamentos de entrada, o compilador cria um arquivo de saída para cada jogador definido no arquivo de entrada podendo o número de arquivos de saída variar de 1 até 5. Estes arquivos de saída do compilador são, na verdade, uma estrutura de objetos, correspondente ao texto de entrada, que descrevem o comportamento dos jogadores (cf. Fig. 3). Desta forma os agentes jogadores (representado pela caixa "Robô" na Fig. 2) podem ler sua especificação de comportamento sem necessidade do processo de compilação a cada execução.

Inicialmente cada agente jogador faz a leitura do arquivo de entrada que contém a descrição do seu comportamento e após se comporta conforme descrito neste arquivo. O agente jogador é criado pelo programa TBSim que faz a leitura do arquivo de descrição do ambiente (cf. Balch (2000)) e, a partir deste, mostra o resultado da execução dos agentes jogador por meio de uma interface gráfica (Fig. 1).

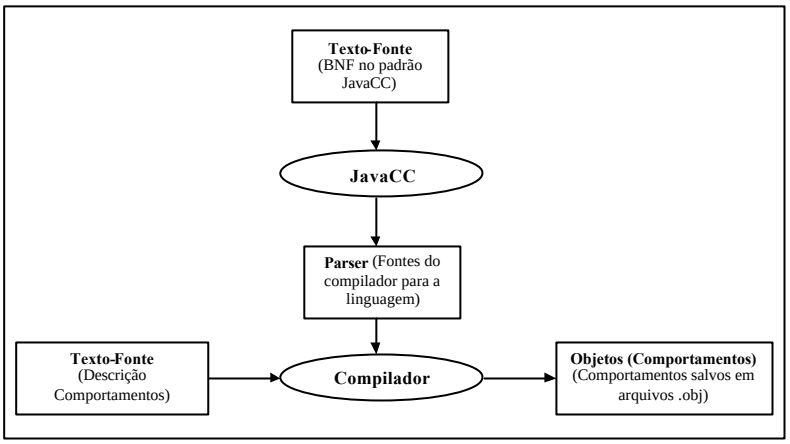

Figura 3: Compilador

4.2 Arquitetura do Agente Jogador

O agente jogador faz comunicação direta com o TBSim (como ilustrado na Fig. 4), sendo assim, toda a percepção do agente é fornecida pelo simulador, com base nesta percepção o agente tem uma memória da situação do ambiente. O módulo de Controle do agente faz a avaliação da memória e ativa o comportamento para a situação encontrada. Após a escolha do comportamento a ser ativado, o módulo executor faz a execução das rotinas (rotina é um conjunto de ações) deste comportamento, a execução das ações primitivas das rotinas são enviadas para o TBSim que simula a execução das ações definidas.

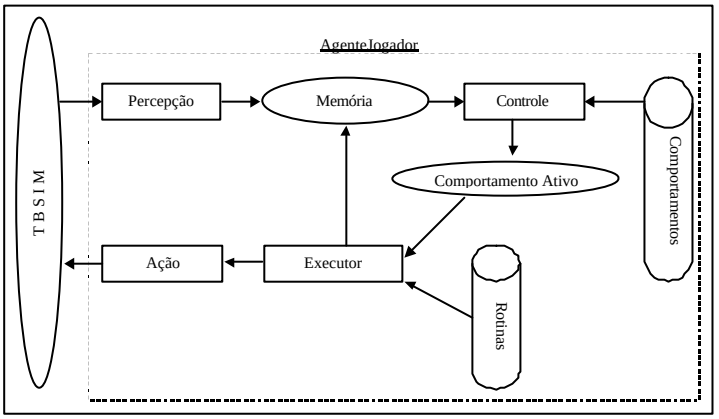

Figura 4: Arquitetura do Agente Jogador

O exemplo apresentado na Fig. 5 descreve a execução do *AgenteJogador*:

a) Caixa Controle: Se a bola está na área de ataque, ativa comportamento Ataque;
   - Comportamento Ataque (executor):
     - se virado pro gol e não tem jogador na frente do gol e pode chutar então (ação) chuta bola pro gol;
     - se virado pro gol e tem jogador na frente do gol então (ação) virar 10 graus à direita e andar.

b) Caixa Controle: Se bola esta na área de Defesa então ativa Comportamento Defesa.
   - Comportamento Defesa (executor):
     - Se jogador adversário com a bola então (ação) ir pra área de defesa;
     - Se jogador com a bola é parceiro então (ação) Parar;

Figura 5: Exemplo de execução do AgenteJogador

Pode-se observar que o controle é formado pelos itens (a) e (b) do exemplo apresentado na Fig. 5, sendo este controle formado por dois tipos de comportamentos distintos, e estes ativados em situações bem definidas. A Fig. 6 mostra a implementação na linguagem proposta do exemplo apresentado na Fig. 5.

```
ControlePrincipal
Inicio
   Se (BolaNaArea("Ataque")) então Ativa(Ataque);
   Se (BolaNaArea("Defesa")) então Ativa(Defesa);
Fim;
Comportamento Ataque
Inicio
   Se ((AnguloDoGol() == 0) e
       (não jogadorNoAngulo(0)) e ComPoseDaBola())
   então Inicio ChutaBola(); fim;
   Se ((AnguloDoGol() == 0) e
       (jogadorNoAngulo(0) e ComPoseDaBola())
   então Inicio GirarParaDireita(10); Andar(1); fim;
Fim;
Comportamento Defesa
Inicio
   Se (AdversarioComPoseDaBola())
   então Inicio IrParaArea("Defesa"); fim;
   Se (ParceiroComPoseDaBola())
   então Inicio Parar(); fim;
Fim;
```

Figura 6: Implementação do Fig. 5

No lugar da descrição formal da sintaxe e semântica da linguagem, será listado aqui um resumo rápido dos principais itens que compõem a linguagem:

a) **dimensaoDoCampo** – este comando criar uma matriz de pequenas seções lógicas sobre o campo de futebol, estas seções existem para facilitar a definição da área na qual o robô vai atuar no campo.

b) **definicaoDoJogador** – este comando indica que vai ser descrito um jogador, sendo o número dele informando como parâmetro, para cada jogador deverá ser descritos 3 seções:

- **areaDeAtuacao** – nomear e definir cada região do campo para este jogador;
- **controlePrincipal** – a rotina de controle principal que faz a escolha do comportamento ativo;
- **comportamento** – a descrição dos comportamentos do agente;

c) **rotina** – este comando define o inicio da descrição de uma rotina, recebendo o nome da rotina como parâmetro.

d) **ações primitivas** – os comandos que representao as ações primitivas do agente são : andar(), parar(), virarParaBola(), chutarBola(), giraParaDireita(), giraParaEsquerda(), iraParaArea(nomeDaArea);

e) **funções numéricas** – as funções numéricas retornam informações do agente em forma de números: distanciaDaBola(), distanciaDoJogador(nr), anguloDoJogador(nr), distanciaDoGol(), anguloDoGol(), limitDoCampo(direita|esquerda|atraz|frente);

f) **funções lógicas** – as funções lógicas retornam informações do agente na forma de verdadeiro ou falso: bolaNaArea(nomeDaArea), comPoseDaBola(), parceiroComPoseDaBola(), jogadorNoAngulo(nr) adversarioComPoseDaBola();

g) **avaliação lógica** – para a utilização das funções tem o comando "se (expressão lógica) então ações" onde no item expressão lógica é utilizado as funções lógicas e numéricas e no item ações são utilizadas as ações primitivas e rotinas;

4.3 Exemplo de uso da linguagem

Esta seção apresenta um exemplo de comportamento que pode ser formalizado com a utilização da linguagem. O exemplo apresentado na Fig. 7 tem a descrição de 5 jogadores, o comportamento dos 5 jogadores faz com que o *AgenteJogador* vá em direção da bola e, após encontrar a bola, chutá-la. Após chutar a bola, continua se movendo em direção a ela e, cada vez que a alcança, chuta-a. A única diferença entre os robôs é a definição da velocidade. A dimensão do campo é de 1x1, assim a definição de área de atuação do jogador para jogo é o campo todo. A expressão *BolaNaArea("jogo")* verifica se a bola esta na área definida no parâmetro sendo verdadeiro ativa o comportamento denominado de "*Comp*". O comportamento define a velocidade que o robô deve se deslocar e faz a

chamada a rotina *"VaiPraBola"*. A rotina *"VaiPraBola"* faz o robô se virar em direção a bola e caso ele alcançar esta, o robô chuta ela.

Utilizando o exemplo apresentado na Fig. 7 é apresentado na Fig. 8 a execução do compilador.

```
DimensaoDoCampo(1,1);
DefinicaoDoJogador ( 1 )
Inicio
  AreaDeAtuacao("jogo",1,1);
  ControlePrincipal
  Inicio
    Se ( BolaNaArea("jogo") )
    Então Ativa (Comp);
  Fim;
  Comportamento Comp
  Inicio
    Andar(0.5);
    chama(VaiPraBola);
  Fim;
Fim;

DefinicaoDoJogador ( 2 )
Inicio
  AreaDeAtuacao("jogo",1,1);
  ControlePrincipal
  Inicio
    Se ( BolaNaArea("jogo") )
    Então Ativa (Comp);
  Fim;
  Comportamento Comp
  Inicio
    Andar(0.8);
    chama(VaiPraBola);
  Fim;
Fim;

DefinicaoDoJogador ( 3 )
Inicio
  AreaDeAtuacao("jogo",1,1);
  ControlePrincipal
  Inicio
    Se ( BolaNaArea("jogo") )
    Então Ativa (Comp);
  Fim;
  Comportamento Comp
  Inicio
    Andar(1);
    chama(VaiPraBola);
  Fim;
Fim;
```

```
DefinicaoDoJogador ( 4 )
Inicio
  AreaDeAtuacao("jogo",1,1);
  ControlePrincipal
  Inicio
    Se ( BolaNaArea("jogo") )
    Então Ativa (Comp);
  Fim;
  Comportamento Comp
  Inicio
    Andar(0.7);
    chama(VaiPraBola);
  Fim;
Fim;

DefinicaoDoJogador ( 5 )
Inicio
  AreaDeAtuacao("jogo",1,1);
  ControlePrincipal
  Inicio
    Se ( BolaNaArea("jogo") )
    Então Ativa (Comp);
  Fim;
  Comportamento Comp
  Inicio
    Andar(0.9);
    chama(VaiPraBola);
  Fim;
Fim;

rotina VaiPraBola
inicio
  VirarParaBola();
  Se ( ComPoseDaBola() ) entao
  Inicio
    ChutarBola();
  fim;
fim;
```

Figura 7: Exemplo de implementação de time de robôs (joga.cmp)

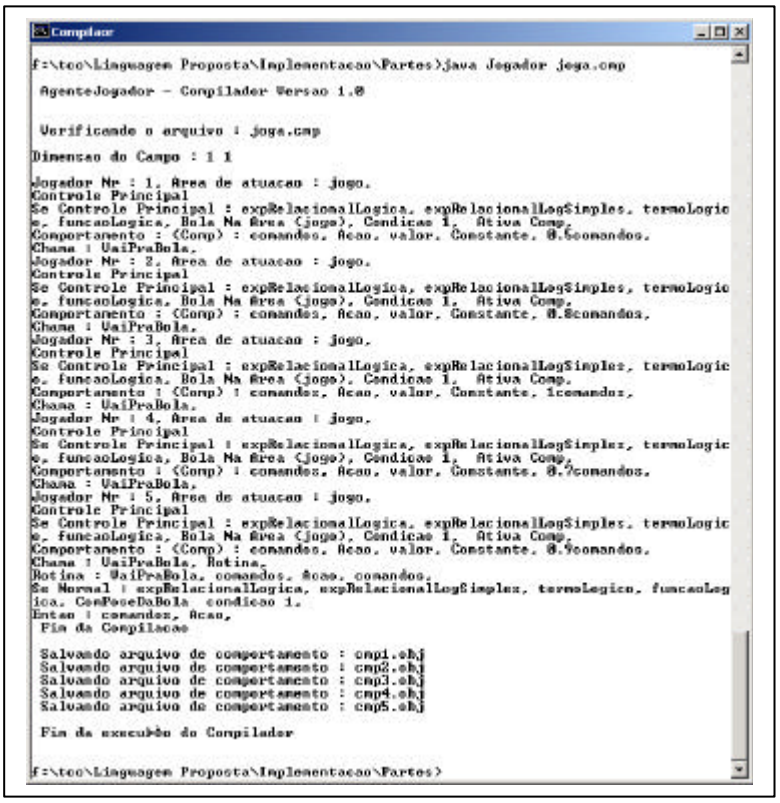

Figura 8: Execução do compilador

O exemplo apresentado na Fig. 7 e a compilação do arquivo de entrada (*joga.cmp*) apresentado na Fig. 8, gerando os arquivos de saída "*CMP1.OBJ*", "*CMP2.OBJ*", "*CMP3.OBJ*", "*CMP4.OBJ*" e "*CMP5.OBJ*". Alterando o arquivo de descrição de ambientes para instanciar somente os robôs da classe *AgenteJogador* no TBSim gera o resultado apresentado na Fig. 9.

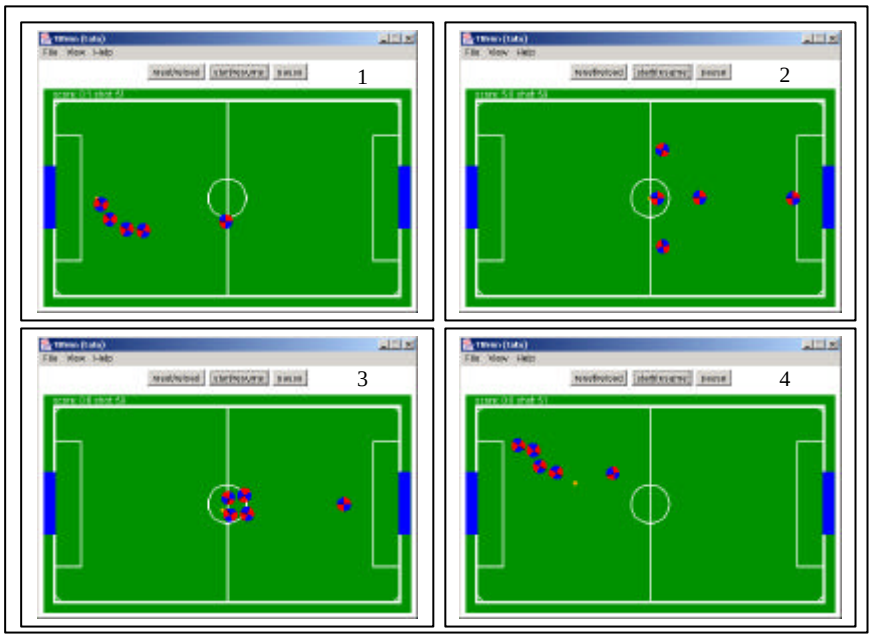

Figura 9: Executando o comportamento

As telas apresentada na Fig. 9 mostra 4 estados diferentes ocorridos durante a simulação utilizando os comportamentos gerados a partir da compilação do exemplo apresentado na Fig. 8. Como o objetivo dos jogadores no exemplo apresentado é de ir para a direção da bola e chutar está independente da direção que esta esteja, podemos então observar que no primeiro estado apresentado tem um jogador com posse da bola antes deste chuta-la, no segundo estado apresentado a bola volta ao centro do campo (isto ocorre quando acontece um gol ou que bola fica parada no campo) e tem um jogador com pose da bola e os outros indo para a direção dela. O terceiro estado apresentado é similar ao segundo só que neste os jogadores estão disputando a posse da bola. No quarto estado podemos observar que um jogador que estava atrasado em relação aos outros jogadores esta indo a direção oposta aos outros e em direção da bola.

5 Conclusão

Sendo o objetivo deste trabalho o de elaborar uma linguagem para descrição de comportamentos de agentes jogadores de futebol no ambiente TeamBots com o propósito de facilitar a implementação e alteração dos comportamentos dos agentes, pode-se concluir que o principal resultado alcançado com este trabalho é uma ferramenta com a qual é possível descrever sistemas de controles para robôs sem a necessidade de conhecer a linguagem Java e a maior parte do ambiente TeamBots.

Referências

BALCH, Tucker; **TeamBots™**, Pittsburgh: [s.n], [200?]. Disponível em: <http://www-2.cs.cmu.edu/~trb/TeamBots/index.html> ou <http://www.teambots.org>. Acesso em: 6 nov. 2001.

BIANCHI, R. A. C. **Uma Arquitetura de Controle Distribuída para um Sistema de Visão Computacional Propositada**. São Paulo, 1998. Disponível em <http://www.lti.pcs.usp.br/~rbianchi/papers.html>. Acesso em 23/04/2002

DAVID, C. Pellejero. *et* al. **FURGBOL - FUTEBOL DE ROBÔS**, Rio Grande - RS, dez. 2001. Disponível em: <http://www.ecomp.furg.br/ecompcricte/2001/Resege36.pdf >. Acesso em: 26 mar. 2002.

JavaCC - Java Compiler Compiler™. **The Java Parser Generator**, Disponível em: <http://www.webgain.com/products/java_cc/ > Acesso em : 1 mar. 2002.

LCMI - Laboratório de Controle e Microinformática. **Estação Ciência**, Florianópolis, maio 1998. Disponível em <www.lcmi.ufsc.br\ufsc-team\ciencia20.html>. Acesso em: 5 nov 2001.

LCMI - Laboratório de Controle e Microinformática. **Departamento de Automação e Sistemas,** Florianópolis, out. 2000. Disponível em: <http://www.lcmi.ufsc.br/ufsc-team/index.htm>. Acesso em: 6 nov. 2001.

ROBOCUP – Robot World Cup. **RoboCup Official Site**, [S.l.: s.n.], out. 2001. Disponível em:<http:// www.robocup.org>. Acesso em: 27 mar. 2002.

SCHLEI, Edson E. **Uma linguagem para definição de estratégias de controle de times de robôs jogares de futebol em um ambiente simulado.** Blumenau, 2002. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) Universidade Regional de Blumenau.