iBots 2010: Descrição do Time

Alexandre Tadeu Rossini da Silva
Universidade Federal
do Tocantins
Email: arossini@gmail.com

Horiano Gomes da Silva
Universidade Federal
do Tocantins
Email: godisgood@uft.edu.br

Edcarllos Gonçalves dos Santos
Universidade Federal
do Tocantins
Email: edcarllossantos@gmail.com

Guilherme Batista Ferreira
Universidade Federal
do Tocantins
Email: guilherme@uft.edu.br

Tanilson Dias dos Santos
Universidade Federal
do Tocantins
Email: tanilsondiasdossantos@gmail.com

Vitor Sousa Silva
Universidade Federal
do Tocantins
Email: vitorsousasilva@gmail.com

***Abstract*— Este artigo apresenta as soluções adotadas pela equipe iBots da categoria Simulation 2D da Robocup. Foi adotado o time-base UvA Trilearn por ter um mapeamento das percepções enviadas pelo Soccer Server e possuir vasta documentação. Foram implementadas e aprimoradas habilidades como pressionar, tocar, chutar, driblar e reposicionar. Como método de tomada de decisão utilizou-se o SARSA, um algoritmo que utiliza o paradigma de aprendizado por reforço e tem como propósito escolher as habilidades a serem executadas por cada um dos agentes em diferentes situações de jogo. O artigo apresenta ainda resultados de testes realizados indicando a viabilidade do SARSA como algoritmo de tomada de decisão na escolha das habilidades a serem executadas pelos agentes no decorrer das partidas. Por fim, são apresentadas indicações de trabalhos futuros a fim de melhorar os resultados obtidos.**

I. Introdução

Este artigo apresenta as soluções propostas e desenvolvidas pela equipe iBots na categoria Robocup Simulation 2D. Foram analisadas as soluções adotadas na implementação de algumas equipes que participaram de competições envolvendo futebol de robôs simulados. Grande parte dos times atuais são construídos a partir de times-base. São vantagens da utilização de um time-base a abstração dos detalhes de comunicação com o simulador, o conjunto de habilidades básicas implementadas e o fácil acesso às informações sobre o estado de jogo.

Essas vantagens permitem que equipes, como o iBots, concentrem-se no desenvolvimento de soluções para problemas como colaboração entre agentes, agentes autônomos, aquisição de estratégias [6]. Há vários times-base disponíveis, porém a maioria com pouca ou nenhuma documentação. Por isso, o iBots foi desenvolvido utilizando-se a equipe UvA Trilearn de 2003 como time-base.

O *UvA Trilearn* surgiu a partir da dissertação de mestrado de dois estudantes de Amsterdã em 2001 [3]. O time-base do UvA Trilearn tem um mapeamento das percepções enviadas pelo Soccer Server e o código fonte é amplamente documentado, motivos bastante convidativos para início do desenvolvimento dos agentes. De acordo com [11], as principais características do time-base do UvA Trilearn são a sincronização flexível entre o agente e o ambiente, os métodos exatos para a estimativa de localização e velocidade de objetos e a hierarquia de habilidades em camadas.

O *UvA Trilearn* é utilizado por importantes equipes do Brasil, como: *Itandrois* [7]; *Mecateam* [2]; e *Bahia 2D* [11].

É apresentada na seção 2 a arquitetura do time-base adotado pelo iBots. Na seção 3 são descritas as habilidades implementadas para os agentes. A seção 4 detalha a tomada de decisão dos agentes a fim de emergir cooperação entre eles. Por fim, a seção 5 discute os resultados parciais obtidos, apresentando seguidamente a seção 6 com os trabalhos futuros a serem desenvolvidos.

II. Arquitetura do time-base

Tarefas complexas, como as do futebol simulado, podem ser hierarquicamente decompostas em sub-tarefas mais simples de modo que a execução dessas sub-tarefas, em uma determinada ordem, resolva a tarefa complexa inicial. Lembrando disto, o agente foi projetado em 3 camadas (vide figura 1),visando eficiência do uso dos recursos computacionais para lidar com o problema de tempo real do simulador. As divisões em camadas também possibilitam que as camadas inferiores executem paralelamente [3].

Em execução, o jogo é observado pelos sensores da **camada de interação** (responsável pela interação com o servidor). Os dados obtidos pelos sensores fluem para a **camada de habilidades**, onde são usadas para atualizar o modelo de mundo e implementar habilidades dos agentes. Uma vez que as informações estejam atualizadas, elas são usadas pela **camada de controle** para escolher a melhor ação, ação esta passada para a camada de habilidades, que tem por objetivo determinar os comandos e os parâmetros a serem enviados para a camada de interação (atuador). Dessa maneira o agente é capaz de perceber, raciocinar e agir [3].

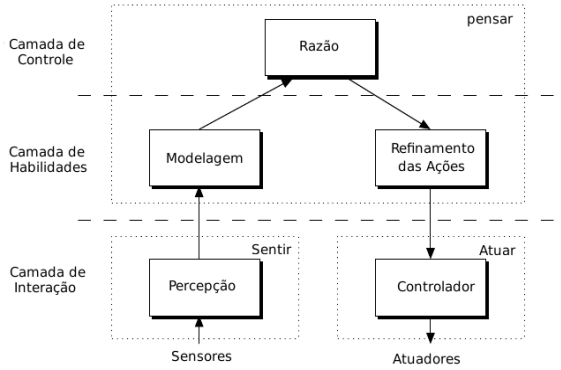

Fig. 1. Arquitetura do Agente. Adaptado de [3].

III. Habilidades

O time-base utilizado tem algumas poucas habilidades implementadas, assim foi necessário, para um melhor desempenho do agente, criar novas habilidades ou alterar as habilidades existentes.

A fim de evitar colisões entre agentes durante a execução das habilidades, o algoritmo de planejamento de trajetória **Campo Potencial Artificial** (Latombe, 1991) foi incorporado às habilidades da equipe iBots. *Campos potenciais artificiais* são muito populares entre os pesquisadores de Robótica e são comumente adotados no futebol de robôs [9] [8]. O princípio básico do campo potencial está em movimentar um robô sob um campo de forças artificiais geradas pelos obstáculos (repulsão) e pelo alvo (atração).

A seguir as habilidades implementadas são apresentadas, todas considerando o nível de qualidade das informações obtidas do modelo de mundo.

A. Pressionar

O método pressionar visa ser utilizado quando o adversário tem a posse de bola e o time quer tentar recuperá-la. Para isso, o agente que executa este método se projeta sobre o vetor calculado entre a bola e o centro do gol da sua equipe. Sobre o vetor, o agente se movimenta em direção à bola no intuito de tentar "roubá-la".

B. Tocar

Este método faz com que um agente passe a bola para um outro agente do seu time. Para isso, é realizada uma análise ponderada considerando alguns fatores presentes no jogo. Os fatores considerados são **segurança da trajetória** (onde são calculadas potenciais ameaças de roubo de bola de agentes adversários durante o passe) e **distância ao gol** (que privilegia os jogadores mais próximo ao gol adversário).

A partir dos fatores citados anteriormente, obtém-se um valor por meio de uma função de avaliação que estipula um valor geral para cada agente analisado de acordo com o peso de cada um dos fatores.

C. Chutar

Este método implementa a ação de chutar. Originalmente o time-base UvA Trilearn tem uma função que chuta para um dos cantos do gol adversário com potência máxima, porém em testes preliminares essa função não se mostrou confiável diante das necessidades da equipe iBots. Assim, foi necessário desenvolver um método que avalia mais detalhes do jogo e considere todo o gol.

D. Driblar

Este método implementa a ação de driblar, para desenvolver foi usado o algoritmo Campo Potencial Artificial para calcular para onde a bola deve ser chutada (com pouca força) para que se consiga manter a posse de bola. Como objetivo de atração foi considerado o centro do gol adversário.

E. Reposicionar

Este método implementa a ação de reposicionar um agente. O time-base já vem com um reposicionar implementado, porém foi necessária a sua modificação uma vez que foram definidas áreas de atuação para cada um dos agentes. O método reposicionar alterado apenas define como ponto de reposicionamento para cada agente o centro da sua área de atuação considerando a posição dos robôs adversários.

IV. Tomada de Decisão

O aprendizado por reforço inclui uma série de algoritmos, onde os mais conhecidas são Q-learning e SARSA. Para a camada de controle foi utilizado o SARSA, que tem convergência mais rápida que o Q-learning [12]. O SARSA foi utilizado para aprender quais das habilidades implementadas deve executar em um determinado momento do jogo.

O SARSA é um processo da cadeia de Markov que considera transição do par estado-ação para aprender uma política ótima. A convergência do método é provada por [12]. A pseudo implementação pode ser vista no algoritmo 1. A implementação do ϵ–greedy é relativamente simples e a pseudo implementação pode ser vista no algoritmo 2. Os valores γ e α são valores que devem ser ajustados com testes empíricos; ϵ é o valor de exploração do agente, quanto maior esse valor mais exploração no domínio o agente fará. Com o tempo o agente deve seguir por caminhos já confiáveis, logo ϵ deve diminuir para que isso aconteça.

Devido ao grande número de estados possíveis para o domínio do futebol de robôs surgiu a necessidade de utilizar um meio para aproximar esses valores. Para o desenvolvimento do algoritmo foi utilizado uma função de aproximação linear chamada de *tile coding*. [12] descrevem o funcionamento e mostram que há convergência com a utilização do *tile coding*.

A vantagem da função de aproximação está em generalizar os estados do jogo, o que permite que a melhor ação (de acordo com sua base de conhecimento) seja escolhida

Algoritmo 1: Sarsa.

Entrada: tabelaValor[*estado*][*acao*]//*Tabela relaciona um valor para um estado e uma ação*
Dados: estado,ação, recompensa, novaAção, novoEstado
estado = o estado atual;
ação = escolha uma ação usando uma politica derivada da tabelaValor;//*exemplo* : $\epsilon - -greedy$
enquanto *estado não é terminal* **faça**
  Execute a ação escolhida
  novoEstado = o novo estado
  recompensa = a recompensa para ação executada
  novaAção = escolha uma ação usando uma politica derivada da tabelaValor; //*exemplo* : $\epsilon - -greedy$
  tabelaValor[*estado*][*acao*] = $tabelaValor[estado][acao] + \alpha(recompensa + \gamma * tabelaValor[novoEstado][novaAcao] - tabelaValor[estado][acao])$;
  estado = novoEstado;
  ação = novaAção;
fim enqto

Algoritmo 2: ϵ–greedy .

Dados: p = um valor aleatório entre 0 e 1
Saída: Uma ação
se $p < \epsilon$ **então**
  retorna *Ação aleatória*;
senão
  retorna *ação de maior valor da tabelaValor dado o estado atual*;
fim se

em estados semelhantes mesmo quando o agente nunca esteve naquele estado.

V. Testes e resultados

Testes foram realizados colocando em disputa a solução implementada pelo iBots contra o Helios-Base [1], por ter um time bem equilibrado, e contra o WrightEagle-Base.

Primeiro foram realizados 100 jogos contra o time HELIOS-BASE, utilizando o sistema tático 4x4x2 e com fator de exploração variando de 99% a 0%, o time não conseguiu marcar gol contra o Helios-Base, porém como pode ser visto na figura 2 o número de gols sofridos diminui conforme o número de jogos aumenta, demostrando que o aprendizado está existindo.

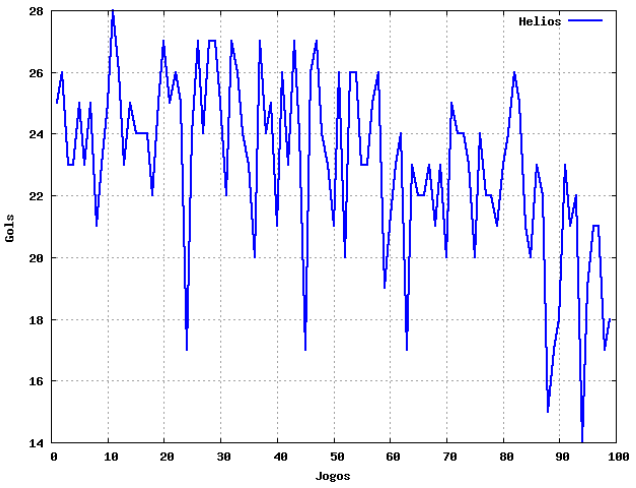

Fig. 2. 100 jogos contra o Helios-Base.

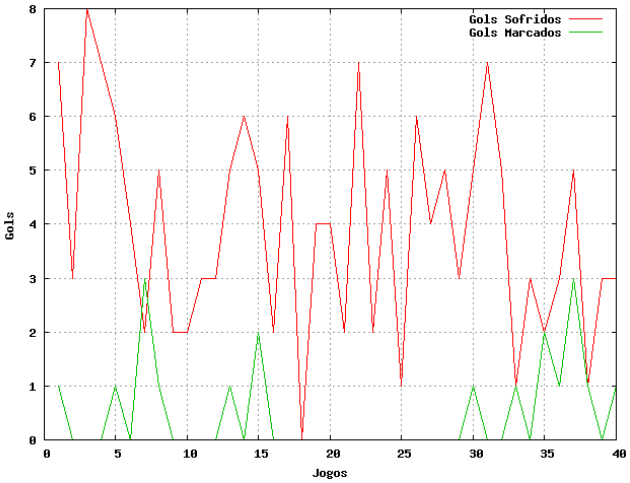

Fig. 3. 40 jogos contra o WrightEagle-Base.

Depois foram realizados mais 40 jogos contra o time WrightEagle-BASE, que é um time mais simples, utilizando o sistema tático 4x3x3 com fator de exploração variando de 10% a 0 % a cada 10 jogos. O resultado pode ser visto na figura 3.

Os testes e resultados permitiram ver que o aprendizado está funcionando, porém ajustes são necessários. Outro fato importante foram falhas identificadas nos métodos desenvolvidos. Assim, estão sendo realizados ajustes nas habilidades utilizadas nos testes. Como principal ajuste está o fato de considerar dois tipos de reposicionamento: ofensivo e defensivo. O reposicionamento descrito na seção 4 deste trabalho está sendo substituído por dois novos métodos:

- **reposicionar ofensivo** - explora a área de atuação do agente de forma discretizada a fim de identificar a posição (dentro da área de atuação do agente) em que se tem maior grau de confiança para receber um possível passe do agente da equipe que está com a bola. Para o cálculo do grau de confiança é utilizado o método de toque, por consequência os fatores considerados são os mesmos do método de toque (descritos anteriormente).
- **reposicionar defensivo** - considera a posição dos agentes adversários dentro da área de atuação do agente e, dentre as posições, o agente identifica o adversário mais próximo ao centro gol da sua equipe para se posicionar entre ele e a bola (considerando um limiar de distância do adversário).

Além disso, foram implementadas novas habilidades que estão em fase de teste:

- **lançar (pass through)** - este método representa uma ação de grande potencial ofensivo. Caracteriza-se por um passe à frente de um jogador, diferentemente do toque que considera a exata posição. Para implementação desta habilidade foi utilizado o método da habilidade toque (descrito anteriormente). As posições consideradas para realizar o lançamento

não são as dos agentes, mas as projetadas sobre o vetor entre o agente em potencial para receber o lançamento e o limite ofensivo de sua área de atuação.

- **marcar** - caracteriza-se em posicionar o agente sobre o vetor entre um adversário definido para ser marcado e o centro do seu gol. Para definir qual adversário marcar, o agente considera todos os jogadores dentro da sua área de atuação e seleciona, dentre os que podem ser marcados por ele (de acordo com sua área de atuação), o que está mais próximo ao gol.

VI. Trabalhos futuros

Atualmente a cooperação entre os agentes ocorre de forma indireta, ou seja, não existe um meio de comunicação entre eles. Assim, cada agente ao tomar uma decisão espera que os demais agentes tomem decisões em consonância com o objetivo da equipe. Por exemplo, no toque, o agente com a posse da bola percebe outro livre e passa a bola esperando que o outro esteja preparado para recebê-la. A comunicação se torna importante para tornar direta a cooperação entre os agentes.

O processo de interação entre agentes, em um time de futebol de robôs, é importante para escolher um melhor conjunto de ações, a fim de atingir o um objetivo comum a todos os agentes, vencer. Com a adição da comunicação, novos conceitos poderão ser agregados como jogadas ensaiadas e troca dinâmica de sistemas táticos durante as partidas. Por isso, a equipe iBots está desenvolvendo mecanismos de comunicação entre os agentes.

Em trabalhos futuros é interessante melhorar as seguintes habilidades dos jogadores: **driblar** (calcular os casos de mínimo local provenientes do Campo Potencial Artificial) e **reposicionar**. Especificamente no reposicionar, avaliar mais detalhes para obter um melhor jogo 'sem bola' decorrente da movimentação dos agentes. Isso visa ocupar de forma mais eficiente os espaços do campo e poupar estamina. A lógica fuzzy tem mostrado resultados bastante interessantes [4].

Ainda pode ser considerada a possibilidade de uso de aprendizado por reforço nas habilidades dos jogadores [10]. Quanto ao aprendizado por reforço é interessante realizar um estudo comparativo do SARSA com o SARSA-λ, que usa traços de elegibilidade.

References

[1] Akiyama, H. 2010 . Helioz - robocup tools, Disponível em: `http://rctools.sourceforge.jp/pukiwiki/index.php?HELIOZ`.

[2] Costa, A. L., Junior, O. V. S. & Teixeira, C. P. 2007 .Tdp Mecateam 2009.

[3] de Boer, R. & Kok, J. 2002 . The incremental development of a synthetic multi-agent system: The uva trilearn 2001 robotic soccer simulation team, Master's thesis, University of Amsterdam, Netherlands.

[4] Garcia, J. M. 2007 . Controladores Fuzzy Para o Posicionamento do Goleiro Em Relação ao Atacante no Futebol de Robôs Simulado 2D.

[5] Latombe, J.C. 1991 . Robot Motion Planning - Kluwer Academic Publishers.

[6] Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I. & Osawa, E. 1997 . Robocup: The robot world cup initiative, AGENTS '97, ACM, New York, NY, USA, pp. 340–347.

[7] Matsuura, J. P., Xavier, R. O. & Barbosa, R. 2006 . O time de futebol simulado itandroids-2d.

[8] Meyer, J., Adolph, R., Sephan, D., Daniel, A., Seekamp, M., Weinert, V. & Visser, U. 2003 . Decision-making and tactical behavior with potential fields, Lecture Notes in Artificial Intelligence - Springer **2752**.

[9] Nagasaka, Y., Murakami, K., Naruse, T., Takahashi, T. & Mori, Y. 2001 . Potential field approach to short term action planning in robocup f180 league, RoboCup 2000, pp. 345-350.

[10] Riedmiller, M., Gabel, T., Trost, F. & Schwegmann, T. 2008 . Brainstormers 2d - team description 2008.

[11] Silva, H., Meyer, J., Oliveira, J., Cruz, D., Pessoa, L., Simões, M. A. C., Aragão, H. & Lima, R. 2007 . Bahia2d: Descrição do time.

[12] Sutton, R. S. & Barto, A. G. 2010 . Reinforcement learning:an introduction, Disponível em: `http://webdocs.cs.ualberta.ca/~sutton/book/ebook/the-book.html`.