UNIVERSIDADE FEDERAL DO AMAZONAS
PRO REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
DEPARTAMENTO DE APOIO A PESQUISA
PROGRAMA INSTITUCIONAL DE INICIAÇÃO CIENTÍFICA

# DESENVOLVIMENTO DE UMA EQUIPE PARA FUTEBOL DE ROBÔS SIMULADOS 2D BASEADOS EM SISTEMAS MULTIAGENTE

Bolsista: Gabriel Gama de Albuquerque

Manaus
2011

UNIVERSIDADE FEDERAL DO AMAZONAS
PRO REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
DEPARTAMENTO DE APOIO A PESQUISA
PROGRAMA INSTITUCIONAL DE INICIAÇÃO CIENTÍFICA

RELATÓRIO FINAL

PIB – E – 0088/2010

# DESENVOLVIMENTO DE UMA EQUIPE PARA FUTEBOL DE ROBÔS SIMULADOS 2D BASEADOS EM SISTEMAS MULTIAGENTE

Bolsista: Gabriel Gama de Albuquerque, CNPq
Orientador: Prof Dr José Francisco de Magalhães Netto

Manaus
2011

# RESUMO

Hoje em dia o Futebol é um esporte muito popular, conhecido e praticado em vários países do mundo, chamando sempre a atenção de pessoas de várias idades. Esse interesse todo traz diferentes modos de usá-lo como objeto de estudo nas mais diversas áreas da Ciência e Tecnologia. A realização de partidas de Futebol acabou ganhando a atenção da Inteligência Artificial junto à Robótica, tornando-se um objeto de incentivo para essas áreas, pois envolve muitos desafios no campo de Sistemas Multiagente, onde a atuação, participação e definição de estratégia em grupo é de maior importância do que seria se fosse focado em individualidades, pois o ponto chave aqui é cooperação dos jogadores (agentes). Este projeto trata do desenvolvimento de uma equipe para futebol de robôs simulados 2D, passando pelas etapas de simulação, implementação e adaptação do sistema Multiagente base do time UvA Trilearn da Universidade de Amsterdam, que disponibiliza somente as habilidades básicas de acordo com as regras da RoboCup Soccer, culminando com a criação de um time base, estimulando o interesse sobre Inteligência Artificial e Robótica, em um projeto de pesquisa motivador.

**Palavras Chaves**: Futebol, Inteligência Artificial, Robótica, Multiagente, RoboCup Soccer.

# ABSTRACT

Nowadays Soccer is a very popular sport known and practiced in various countries around the world, always calling the attention of people of various ages. This interest brings all different ways to use it as an object of study in various fields of Science and Technology. The realization of Soccer matches ended up winning the attention of the Robotics with Artificial Intelligence, becoming an object of incentives to these areas, because it involves many challenges in the field of Multiagent systems, where performance, participation and strategy definition in group is of greater importance than it would if it were focused on individuality, because the key point here is cooperation of the players (agents).This project is developing a robot soccer team for simulated 2D, passing through the stages of simulation, implementation (Multiagent system based on the team Trilearn UvA, University of Amsterdam, which provides only basic skills in accordance with the rules of RoboCup Soccer) and creating a team based, stimulating interest in Artificial Intelligence and Robotics, in a research project motivator.



**Key Words**: Soccer, Artificial Intelligence, Robotics, Multiagent, RoboCup Soccer.

# LISTA DE FIGURAS

# SUMÁRIO

# 1 INTRODUÇÃO

A Inteligência Artificial (IA), área de pesquisa da Ciência da Computação e Engenharia da Computação, é dedicada a buscar métodos ou dispositivos computacionais que possuam ou simulem a capacidade racional de resolver problemas, de uma forma sempre significativa e inteligente.

Com o artigo “Computing Machinery and Intelligence” de Alan Turing começou o desenvolvimento dessa área, junto com muito outros cientistas idealizadores. A IA já possui muitos campos onde é aplicada: Planejamento Automatizado e Escalonamento, Controles Autônomos, Programas de Diagnósticos, Robótica, Reconhecimentos, entre outros. Os campos de maior importância aqui para nós é o de Jogos e Robótica, onde se aplica o Projeto de Futebol de robôs Simulado 2D, utilizando sistema multiagente, onde os jogadores são agentes inteligentes (nesse caso, entidades lógicas e não físicas por tratar-se de um ambiente simulado) agindo em prol de um objetivo em comum.

Existindo hoje competições de nível estadual e até mundial como a RoboCup, sendo um meio de desafio e uma forma de prática nas soluções de problemas na área de computação, tem-se alcançado resultados significativos de ter cada vez mais adeptos e simpatizantes desse tipo de estudo/entreterimento que só faz favorecer ainda mais descobertas e alcançar resultados.

# 2 REVISÃO BIBLIOGRÁFICA

Alguns conceitos e Termos precisam ser entendidos para compreensão do trabalho, dentre eles: Inteligência Artificial Distribuída, Sistema Multiagente, RoboCup Soccer.

## 2.1 Inteligência Artificial Distribuída

A Inteligência Artificial Distribuída (IAD) é uma subárea da Inteligência Artificial (IA), um dos mais promissores ramos da Ciência da Computação, quase sempre caminhando junto à robótica. A IAD tem dois grupos de modo de resolução de problemas, um que foca nas soluções e outro que foca nas entidades do problema, são eles respectivamente: Resolução Distribuída de Problemas (RDP) e o Sistema Multiagente(SMA). Ambos usando os termos de “agentes”, agentes autônomos que estão situados em um ambiente, no qual eles sentem e agem, sobre o tempo, mantendo a sua própria agenda e sentem os efeitos de suas ações no futuro (Franklin e Graesser,1996,p.5), na resolução de problemas.

## 2.2 Sistema Multiagente

O Sistema Multiagente é um sistema computacional onde existem dois ou mais agentes que interagem e comunicam entre si, agindo de forma autônoma mas sempre em busca de um objetivo ou uma resolução de problema em comum [Lesser,1999]. A característica mais notável nesse sistema é a coletividade que é atingida utilizando os agentes de formas independentes, onde cada um pode desempenhar o seu papel e se comunicar para que haja um entendimento entre o grupo e o sucesso seja alcançado.

## 2.3 RoboCup Soccer

A RoboCup Soccer é uma das modalidades da RoboCup, um torneio que visa divulgar e aumentar o interesse por Inteligência Artificial e Robótica. Essa modalidade traz basicamente um problema, que é jogar futebol, onde 11 agentes para cada uma das duas equipes são colocados em um campo de futebol virtual de modo a interagir como em um jogo de futebol  e quem tiver o maior número de gols em um tempo

Regras do servidor Soccerserver que controla e simula a partida são aplicadas, os jogadores constituem os clientes e os problemas são aplicados na prática: processamento, comunicação, cooperação e outros relacionados ao esquema Servidor-Cliente ou de maneira mais popular, campo-jogadores.

A RoboCup Soccer traz um estimulante desafio científico para a área de Inteligência Artificial, pois vem com problemas a serem estudados e soluções sempre a serem implementadas.

.

# 2 MÉTODOS UTILIZADOS

No decorrer do projeto foram realizadas diversas pesquisas bibliográficas referentes ao Sistema Operacional Linux e seus comandos, Sistema Multiagente, Futebol 2D Simulado na Robocup Soccer e suas regras, Instalação do Simulador de Partidas de Futebol 2D Simulado, e Ambiente de Programação ANJUTA 2.32.1.1.

A pesquisa sobre o Sistema Operacional Linux serviu para adquirir conhecimentos sobre o modo de instalação de pacotes pelo Gerenciador de pacotes ou pelo terminal, como compilar um arquivo dado uma distribuição qualquer Linux e alguns comandos padrões que são necessários para poder dar continuidade ao projeto nesse Sistema.

Futebol 2D simulado na Robocup e suas regras foi o tema principal das pesquisas envolvendo desde a pesquisa sobre o que era e como funcionava futebol 2D simulado, o que é a RoboCup e como funciona a modalidade RoboCup Soccer e suas regras.

A instalação dos pacotes que fazem a Simulação do campo, dos agentes (jogadores) e pequeno Player (reprodutor de vídeos) que roda os arquivos de .LOG ( arquivos com informação da partida e tudo que aconteceu nela) para poder ser analisada a fim de definir estratégias.

O Ambiente de Desenvolvimento Integrado para as linguagens C e C++ em GNU/Linux escolhido pra alteração e implementação do Código-Fonte do Time UvA Trilearn foi o ANJUTA 2.32.1.1 pois suporta muitas capacidades avançadas como gerenciamento de projetos, importante nessa caso pois o código do programa que faz o time possui vários arquivos, e um poderoso editor de código fonte.

# 3 RESULTADOS E DISCUSSÕES

A instalação dos pacotes do simulador na plataforma Linux Ubuntu 11.04 e o modo como utilizá-lo foi uma das partes mais importantes, então deve ser descrita adiante para entendimento caso exista alguma dificuldade nessa parte, os testes feitos usando times bases também e a instalação do ambiente de programação.

A arquitetura básica dos componentes que estão no projeto é a seguinte:

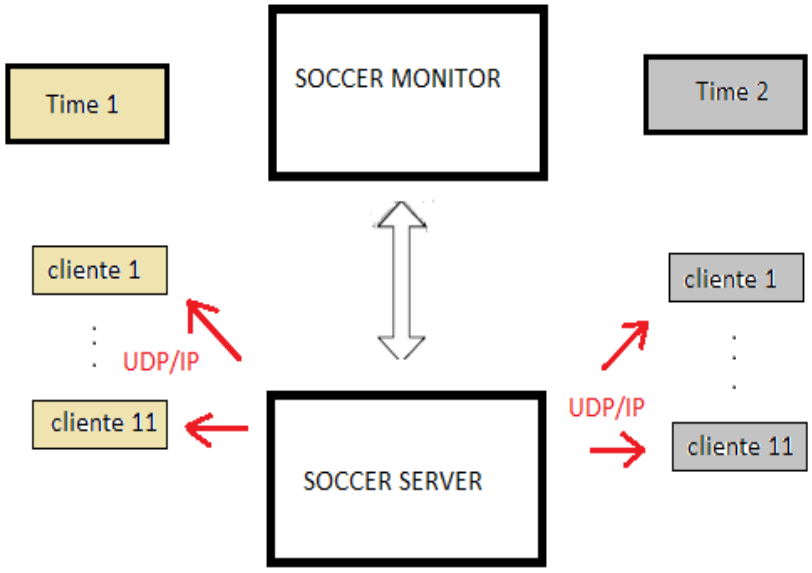


Figura 1: Arquitetura de componentes envolvidos na simulação

Onde o Servidor (Pacote de instalação rcssserver) realiza a simulação de jogos, o Monitor (Pacote de instalação rcssmonitor) permite a visualização dos jogos do simulador e o vídeo (Pacote de instalação rcsslogplayer) pode reproduzir gravação das partidas realizadas utilizando arquivos de Log.

Antes de instalar os Pacotes pré-compilados para a distribuição DEBIAN 32bits (e baseadas nela, como o UBUNTU) deve-se instalar algumas Bibliotecas (Dependências) para que rode sem nenhum problema o simulador.

Opta-se aqui por instalar as bibliotecas pelo modo de gerenciador de pacotes, o Synaptic. Passo para chegar até lá basicamente é assim: Menu → Sistema → Gerenciador de Pacotes, clique em Recarregar para atualizar, aí no espaço de busca

ou pesquisa digite o nome do pacote desejado, são eles: libltdl7, libstdc++6, libboost-filesystem1.40.0, libboost-program-options1.40.0, libqtcore4, libqtgui4, libqt4-opengl, libqt4-network, libxaw7, libxpm4, libxt6, gcc, g++, tcsh.

No caso de ter alguma versão diferente de uma das libboost citadas anteriormente será necessário baixar a versão pedida aqui, pois o simulador não irá funcionar se for alguma versão diferente dessas, indo no site do repositório do Ubuntu em libboost-filesystem1.40.0 [LIBBOST-SYSTEM1.40.0] e em filesystem [LIBBOST-FILESYSTEM1.40.0] e também a libboost-program-options1.40.0. [PROGRAM-OPTIONS].

Depois de instaladas as dependências [DEPENDENCIAS] a próxima coisa a fazer é instalar o simulador com os pacotes: rcssserver, rcssmonitor, rcsslogplayer, que podem ser obtidos na página da internet, site que possui todas as versões dos pacotes do simulador. As versões dos pacotes pré-compilados instalados foram rcssserver_14.0.3-1,rcssmonitor_14.1.10-1,rcsslogplayer_14.0.1-1.

Para instalar colocamos os pacotes .deb em uma mesma pasta,abrimos o terminal Linux, navegamos até aonde está a pasta, e como root(administrador) executa-se o comando: dpkg –i *.deb. Pronto, o simulador vai ser instalado.

Os times a serem compilados para teste foram: Time UvA Trilearn Base [UVA TRILEARN] da Universidade de Amsterdam e um outro time base de mesmo código com poucas alterações, o Time UvA Trilearn Base_modificado [TIME BASE UVA_TRILEARN_2003 ATUALIZADO] pelo Pet de Engenharia da Computação da UFES (Universidade Federal do Espirito Santo).

Para compilar um time, descompactar a pasta, abri-se um terminal, navegar até a pasta e executar o comando: *./configure*, e logo em seguida depois de ser verificado se as dependências estão todas instaladas e criar um Makefile do time, digita-se e executa: *make*. Após isso o time está compilado e pronto para uso.

Na execução do simulador basta abrir um terminal e executar o comando: *rcsoccersim. Abrirá o simulador como na Figura 2.*


Figura 2: Tela do simulador

Para carregar o primeiro time (Figura 3) é só navegar até a pasta onde se encontra com outro terminal aberto e digitar e executar o comando: *./start.sh.* Para um segundo time (Figura 4) também é só navegar até a pasta pelo terminal e digitar: *./start.sh localhost time2,* sendo que *time2* é o nome do time que aparece no monitor, esse nome tem que ser diferente do nome do primeiro time se não o simulador entende que são mais de 11 jogadores no mesmo time.


Figura 3: Entrada do primeiro time em campo

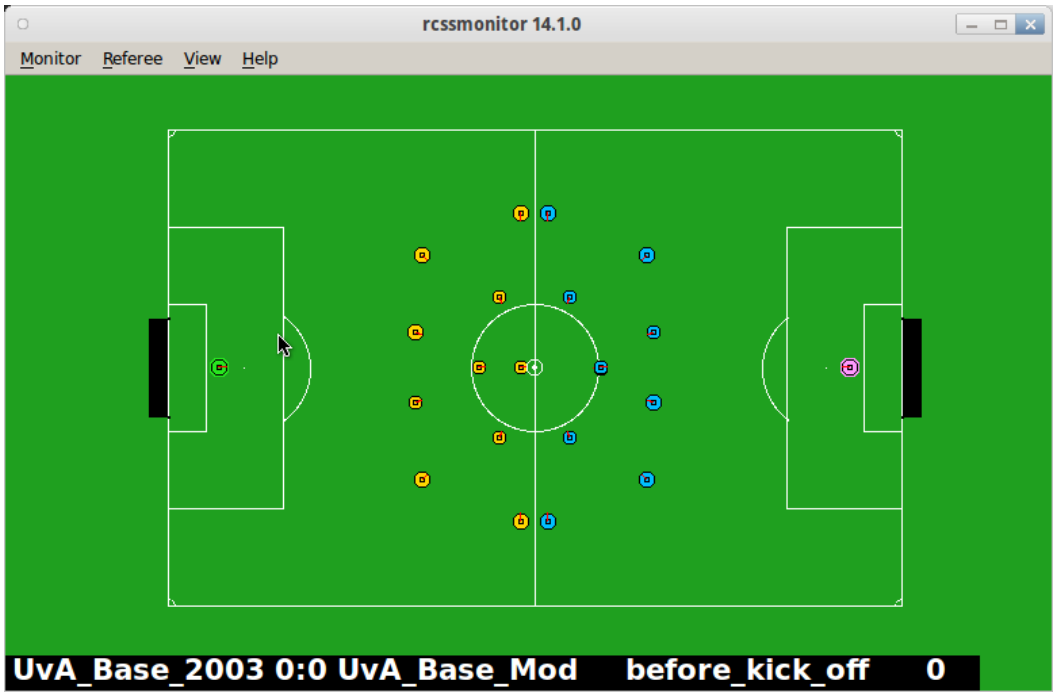


Figura 4: Entrada do segundo time em campo

Para iniciar a partida (Figura 5) depois que todos os jogadores entram em campo, é só apertar em Referee->kickoff no menu do simulador. Assim que o primeiro tempo acabar ele dá uma pausa, então deve ser apertado o novamente o botão para começar o próximo período de ciclos. O simulador possui uma tabela de atalhos em Help->Shortcut Keys, onde pode dar uma noção melhor de pequenas configurações do simulador.

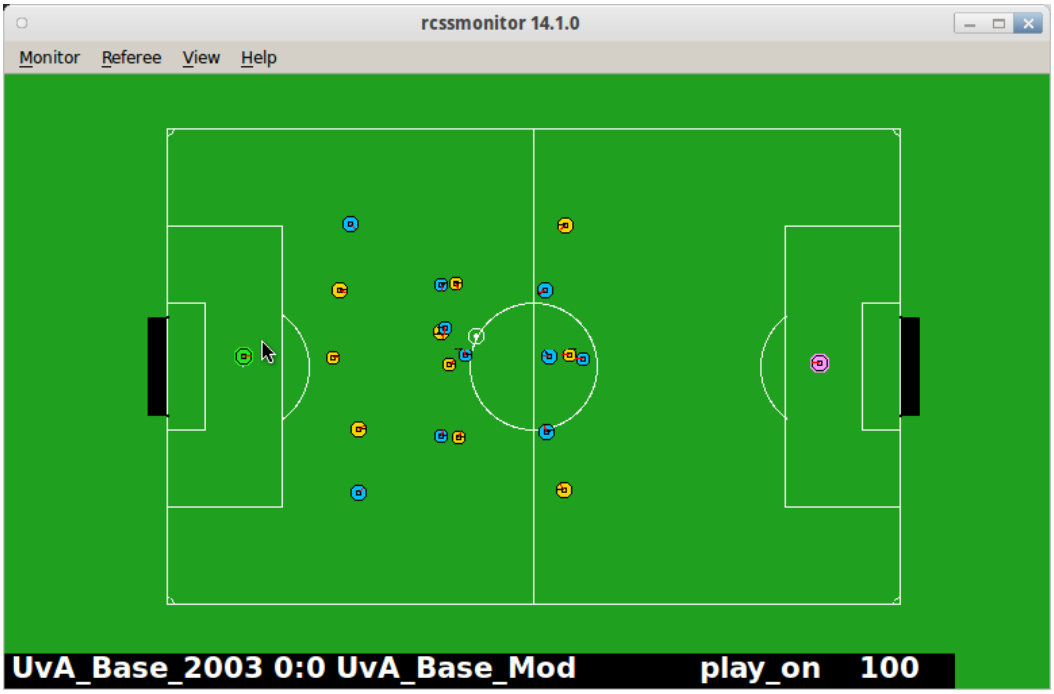


Figura 5: Realização de uma partida

# 5 TESTES E AVALIAÇÕES

Para trabalhar com programas maiores e com vários arquivos diferentes optou-se pela escolha de um ambiente de programação mais completo, a IDE ANJUTA na versão 2.32.1.1 foi instalada. Depois da instalação ao abrirmos a IDE, importamos o time com a opção: “projeto a partir de código existente”, coloca-se um nome para o projeto e selecionamos a pasta onde está o time base. Na opção arquivos que apareceu após o projeto ser importado podemos visualizar todos os arquivos do projeto. Entre os dois editores presentes, o usado aqui é o Scintilla, mas não há nada que impeça de usar o GtksourceView. O Anjuta ainda possui opções para autocompletar, plug-ins e configurações adicionais, e outras que ficam a critério do programador como preferência na sua forma de trabalho.

No diretório “SCR”, Arquivo PlayerTeams.cpp do time base tem uma função SoccerCommandPlayer::deMeer5(),linha 60, que é onde deve a princípio ser implantado comandos básicos de decisões do jogador chamando funções específicas como de chutar, driblar ou dar um passe. A função DeMee5_goalie também segue o mesmo raciocínio mas com comandos e decisões(funções chamadas) diferentes. A seguir um trecho de SoccerCommandPlayer::deMeer5() :

```
 if( WM->isBallKickable() )
 {
      VecPosition posGoal( PITCH_LENGTH/2.0,
                           (-1 + 2*(WM->getCurrentCycle()%2)) *
                           0.4 * SS->getGoalWidth() );
      soc = kickTo( posGoal, SS->getBallSpeedMax() );
      Log.log( 100, "take kick off" );
 }
     else
{
      soc = intercept( false );
      Log.log( 100, "move to ball to take kick-off" );
 }
     ACT->putCommandInQueue( soc );
     ACT->putCommandInQueue( turnNeckToObject( OBJECT_BALL, soc ) );
         return soc;
```

No código acima a função tem como estratégia inicial sempre chutar ao gol quando a bola estiver na área de chute do jogador.

Mudanças na estratégia utilizando apenas algumas funções já prontas que se encontram no arquivo BasicPlayer.cpp do time base foram feitas:

- CHUTE - Como ideia inicial não utilizar toda potência do chute toda vez que essa função é chamada, esperando assim obter como resultado menos stamina(força, energia) desnecessariamente consumida por um jogador. Mudar coordenadas para que o jogador chute obrigatoriamente a partir desse ponto quando estiver com a bola em direção a outra coordenada qualquer, utilizando funções getX() e getY(). Por exemplo:

```
if (posAgent.getX() < 40)
{
  soc = kickTo ( VecPosition (35,10), 0.5);
}
```

- DRIBLE – função para carregar a bola e basicamente dá um chute de leve pra o ângulo que for determinado:

```
soc = dribble (0,DRIBBLE_SLOW);
```

Os testes alcançaram resultados parciais com desempenhos muito irregulares, sendo impossível atestar de forma convicta a melhora ou piora da equipe modificada. Devido à exiguidade do tempo não foi possível concluir o projeto em seu objetivo principal de maneira satisfatória, assim não chegando à fase de implementação do código no que se refere à criação de novas funções.

# 6 CONCLUSÕES

Por motivo de desistência do bolsista anterior, este trabalho apresenta os resultados de cerca de três meses de trabalho, conforme enfatizado na apresentação do CONIC. Embora o objetivo principal do projeto de criar uma equipe de futebol de robôs simulados 2D não tenha sido totalmente alcançado devido à exiguidade do tempo, objetivos secundários como a instalação do simulador, e os testes realizados entre equipes de times bases da Universidade de Amsterdam e o time base que a UFES disponibiliza com pequenas modificações em relação ao primeiro, foram realizados, além da elaboração de protótipos.

O simulador e todo o processo que envolve sua execução foi instalado e compreendido com êxito, resultando no bom funcionamento e evitando problemas de grande e pequeno porte nas partidas simuladas.

Os códigos-fonte dos programas dos times podem ser estudados e implementados a fim de testes em testes obter um resultado cada vez melhor a favor dos times modificados que se sobressaem.

O Ambiente de programação Anjuta mostra ser um grande auxiliador para este tipo de projeto que possui vários arquivos de bibliotecas que são indispensáveis para a implementação do código-fonte do time.

# 7 CRONOGRAMA

| Nº | Descrição | Ago 2009 | Set | Out | Nov | Dez | Jan 2010 | Fev | Mar | Abr | Mai | Jun | Jul |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **1** | Pesquisa Bibliográfica e Webgráfica | **o** | **o** | **o** | | | | | | | | **X** | |
| **2** | - Instalação de Ferramentas de Apoio (Simulador RoboCup e outros); | | | **o** | **o** | | | | | | | **X** | |
| **3** | Levantamento de Sistemas Disponibilizados (Software de Equipes); | | | **o** | **o** | **o** | **o** | | | | | **X** | |
| **4** | Projeto, implementação, testes e avaliação comparativa de protótipos e sistemas disponibilizados | | | | | **o** | **o** | | **o** | **o** | | | |
| **5** | Escrita de um relatório do projeto | | | | | | | | | **o** | **o** | **o** | **X** |
| **6** | -<br>- Elaboração do Resumo e Relatório Final (atividade obrigatória)<br>- Preparação da Apresentação Final para o Congresso (atividade obrigatória) | | | | | | | | | | | | **X** |

**Legendas :**

**X – Atividades realizadas por mim**

**o – Atividades realizadas por outro bolsista**



# REFERÊNCIAS BIBLIOGRÁFICAS

[DEPENDENCIAS]. Disponível em: < http://sourceforge.net/projects/sserver/files/>. Acesso em: 22 de junho de 2011.

[SIMULAÇÃO 2D]. Disponível em: <http://www.inf.ufes.br/~pet/projetos/Simulacao_2D/Simulacao2D.html>. Acesso em: 15 de junho de 2011.

[FILESYSTEM]. Disponível em:< http:// packages.ubuntu.com/maverick/libs/libbost-filesystem1.40.0>. Acesso em: 28 de junho de 2011.

[FRANKLIN E GRAESSER,1996] Franklin, S. e Graesser, A. Is it a n Agent, or just a Program?: A Taxonomy for Autonomous Agents, Proceedings of the Third International Workshop on Agent Theories, Springer-Verlag, 1996.

[LESSER,1999] Lesser, V.R., 1999. Cooperative multiagent systems: A personal view of the state of the art. IEEE Trans. Knowledge Data Eng.

[OXENTE, 2010] Oxente team: implementação de habilidades de jogo em um time de futebol de robôs simulado utilizando otimização heurística. Disponível em: < https://intranet.dcc.ufba.br/pastas/mecateam/material_de_estudo/artigos/artigoOxente.pdf>. Acesso em: 22 de junho de 2011.

[PROGRAM-OPTIONS]. Disponível em: < http:// packages.ubuntu.com/maverick/libs/libbost-program-options1.40.0 >. Acesso em: Acesso em: 28 de junho de 2011.

[REIS, 2003] Reis, Luís Paulo. Coordenação em Sistemas Multi-Agente: Aplicações na Gestão Universitária e Futebol Robótico ( Coordenation in Multi-Agent Systems: Applications in University Management and Robotic Soccer), PhD Thesis, FEUP, july of 2003, pág 49-82. Disponível em: < http://paginas.fe.up.pt/~lpreis/Research.htm#Tese de Doutoramento >. Acesso em: 22 de junho de 2011.

[SYSTEM]. Disponível em: < http://packages.ubuntu.com/maverick/libs/libbost-system1.40.0>. Acesso em: 28 de junho de 2011.

[TIME BASE UVA_TRILEARN_2003 ATUALIZADO]. Disponível em: <http://www.inf.ufes.br/~pet/projetos/Simulacao_2D/Sim2D/Arquivos/trilearn_base_sources-3.3-v14-by_PET.tar.gz>. Acesso em: 28 de junho de 2011.



[UVA TRILEARN BASE]. Disponível em: < http://staff.science.uva.nl/~jellekok/robocup/2003/>. Acesso em: 28 de junho de 2011.