An Introduction to
RoboCup
and
Soccer Simulation 2D

February 28, 2009

Danial Khashabi
Amirkabir University of Technology
Overview of lecture

- Overview of RoboCup
- Necessity of Simulation Leagues
- Overview of 3D Simulation
- Simulator Mechanics
- Simulator Communications Protocol
- Some Existing Clients
- Other Developments/Tools
What is RoboCup?!  

- The Robot World Cup Initiative  
- Started in 1992 as the Robot J-League (Japan)  
- First games and conferences in 1997  
- Workshops, conferences and yearly competitions  
- Slogan of games:  
  “By the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team.”
The Goal of Robocup?

- RoboCup envisions a set of longer range challenges over next few decades
- A standard problem for research in:
  - Artificial Intelligence and Machine Learning
  - Machine Vision and Image Processing
  - Natural Language Processing
  - Multi-Agent Team Planning
  - And different challenges in Control and Electronics and Computer Science...
### RoboCup Soccer Leagues

<table>
<thead>
<tr>
<th>Standard Platform</th>
<th>Small Size</th>
<th>Middle Size</th>
<th>Simulation</th>
</tr>
</thead>
</table>
Why Simulation?!

- Since it is not likely that the ultimate RoboCup goal will be met in the near future, it is important to also look for short-term objectives.
- The fact that no expensive hardware is needed to build the team.
- It provides a standard problem for the evaluation of various theories, algorithms and architectures.
- It abstracts from hardware issues and focuses on subjects such as skill learning, coordination techniques, and opponent modeling.
- It is much easier (and cheaper) to test a simulation team against different opponents.
Why Soccer?!  
The most popular sport in the world  
Two teams ↔ Antagonism  
11 players each ↔ Distributed  
Complexity ↔ having a lot of conditions  
Uncertainty ↔ Result of chaotic behaviour

RoboCup Soccer Simulation  
A research and educational project for multi-agent systems and artificial intelligence.  
The attempt to model a real-life situation on a computer so that it can be studied to see how the system works.
<table>
<thead>
<tr>
<th>2D Simulation</th>
<th>3D Simulation</th>
</tr>
</thead>
</table>

RoboCup Football Simulation Leagues
RoboCup Simulation

Consists of:

- Server
- Monitor clients
- Player clients (i.e. agents!)
- Coach clients for each team
2D Soccer Simulation

Two teams of 11 virtual agents each play with each other.

Based on a computer simulator (i.e. Soccer Server or RCSSServer) that provides a realistic simulation of soccer robot.

One demonstrator software to show processes in graphical interface (i.e. Soccer Monitor or RCSSMonitor).

Each agent is a separate process that sends the simulation server.

Environmental information are sent to each agent, after actions’ occurrence based on noise values.

Why noise?
Realistic
Uncertainty

....
What is different from computer games?!

Goals

entertainment! Vs. research and education

Differences in characteristics:

Reality
None-deterministic
Noisy
Partially observation
Continuous
Real time
Distributed
Practical
Monitor(s) used to visualize the action and/or interrupt the game

Coaches (optional) to give guidance to teams

One server

Up to 11 clients per team (plus one coach)

Clients/server communicate via UDP/IP

TCP/IP support will be added soon!
Soccer Agents

A cycle between server and clients:
- Sensory information received from server, decision made, action command sent back to server
- Clients may talk only to the server... not to each other!
- Clients’ connection is just possible by restricted use of “Say Command”
- Clients; Autonomous agents
- One client represents one player, goalie or coach
- Clients can be written in any language (C++, Java, Smalltalk, ...)
- Clients can be run on same machine or a network
Clients

An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors.

The outcome of actions is non-deterministic, consequently the agent has to learn a model for their prediction.

Observations are noisy and ambiguous, consequently the agent has to build an internal representation of the world, known as a “world model”.

The environment is dynamic, consequently the agent has to interact in real time.

The environment contains more than one agent, consequently the agent has to compete or cooperate with other agents.
RoboCup Clients

Surrounding Info:
- time
- play mode
- my body
- landmarks
- other players
- (Say)speech
- coach instructions

Input from server

Player Agent
- Parsing new info
- And
- decision-making

Output to server

T(n)

Actions:
- dash
- turn
- turn head
- kick
- catch
- speak

T(n+1)
RoboCup Soccer Server

- Keeps time (typically 6000 simulator cycles, 10 cycles per second)
- Sends new status information
- Receives client messages and updates “world model”
- “Automated Referee” tracks current play mode
Coach Agent

- Privileged clients used to provide assistance
- Receives noise-free view of the whole field
- Used for opponent modeling, game analysis, giving strategic tips to teammates

1- Online Coach: Used to advise players during game

2- Offline Coach(Trainer): Used to train agent’s in out of game time. This is an important tool for implement AI algorithms on one team.
Some RoboCup Source Codes

- Mersad (Iran)
- WrightEagle_BASE (China)
- UvA Trilearn (Amsterdam) (2003 champion)
- HELIOS (Japan)
- Agent2D (HELIOS_BASE) (Japan)
Typical Approaches

- Hard-coding (i.e. trial and error !)
- Layered Learning
- Reinforcement Learning
- Artificial Neural Networks
- Markov Decision Process
- Fuzzy methods
- ...
- and combination of items above with together .

Focus of implementations are also different :
- Decision Making problem. (Cooperative, MAS)
- Skill modeling (Individual or dual )
- Opponent Modeling(e.g. about specific skill, cooperative task or positioning )
- ....
Example: Krislet

- Only one strategy: run to the ball and try to kick it!
- Surprisingly effective: Some times it a good challenge for trained teams to test their algorithm’s effectiveness versus a stupid team!
- Written in Java, easy to extend
Example: Stripslet

- STRIPS style linear planning
- Written by Aloke Wiki
- Based off of Krislet
- A Stripslet implementation is made up of four main concepts: Actors, Sensors, Actions, and a GoalList
Example: UvA Trilearn

- Coordination Graphs for cooperative tasks e.g. passing, anticipating passing points
- Layered skills hierarchy (e.g. pass and intercept)
Example: FC Portugal

- Strategic, ball possession, ball recovery behaviours
- “Situation Based Strategic Positioning”
- Given game situation, calculated best position and go there
Example: Brainstormers

- Scientific research Vs. competitions
- Neural Modeling; Most off skills are written based on MLP.
- MDP-based decision making
- The best team in history of 2d league
Example: WrightEagle

- Neural Modeling
- Probabilistic approach to decision making
- Great offline noise modeling
Example: HELIOS

- Most of team is hard coded and is result of experimental tests.
- The most heuristic part of this team is its semi-dynamic base positioning, based on Delaunay Triangulation. A local and fast function approximation.
Research On Soccer Simulation?!

Try to construct agents that are both autonomous and rational. Autonomous implies that the agent makes its decisions without the guidance of a user.

Rational means that the agent selects those actions that are expected to achieve the best expected outcome based on the available information.
Research

Opponent Modeling

Modeling and reasoning about other agent’s goals, plans, knowledge, capabilities, or emotions — is a key issue in multi-agent interaction.

On-line tracking  Involves individual players’ real-time, dynamic tracking of opponents’ goals and intentions based on observations of actions.

On-line strategy recognition  ”Coach” agents for teams may observe a game from the sidelines, and understand the high-level strategies employed by the opposing team.

Off-line review  ”Expert” agents may observe the teams playing in an after-action review, to recognize the strengths and weaknesses of the teams, and provide an expert commentary.
How do I join RoboCup?!

Knowledge

Programming Language: C++ (STL, boost), Script (bash, MATLAB, python...)
Linux Operation
Computer Algorithms, Artificial intelligence, ...
Software Engineering
Starting a Game

- Download and install applications:
  - RCSSServer: Main game running program. Teams connect to this module.
  - RCSSMonitor: Visualiser of game.
  - RCSSLogplayer: A tool for reviewing post game.
    - (running the configure and make scripts for the Unix / Linux systems)

- Run the Server (default host is localhost and default port is 6000)
- Run the Monitor, connecting to the host and port of the Server
- Connect the players to the Server host and port
- Start the game!
Read More?!

- About RoboCup:  www.robocup.org
- Soccer Server Manual
- Robosina from scratch (Bachelor Thesis)
- Articles and TDPs of world cup teams
How To Start?!

Start with UvA-Trilearn-Base:

- Download: http://staff.science.uva.nl/~jellekok/robocup/2003/

Compile it with `configure; make;`

We will analyze different aspects of UvA-Trilearn-Base code next section
Most of the information about RoboCup itself was taken using from the RoboCup Soccer Server manual.

For the latest manuals and server versions, visit the RoboCup project website at: http://sourceforge.net/projects/sserver