CEC 2007 Tutorial
September 25, 2007, Singapore

Evolving Soccer Teams for RoboCup Simulation

Tomoharu Nakashima
Osaka Prefecture University, Japan
nakashi@cs.osakafu-u.ac.jp

Outline

- Part-I
  - Soccer Simulation
  - Simulation models
- Part-II CI for RoboCup Simulation
  - Fuzzy Systems for Ball Intercept
  - Neural Networks for Mimicking Dribble
  - Evolutionary Computation for Team Strategy

RoboCup Soccer

RoboCup Challenge

First RoboCup competition (1997)

About 50 years

By the year 2050, develop a team of fully autonomous humanoid robots that can win against the human world soccer champion team

Moon Landing by Human Beings

First flight of aircraft (1903)

Apollo program (1969)

About 60 years

Computer Chess Challenge

Digital computers (1950's)

Deep Blue won against Kasparov (1997)

About 50 years

RoboCup Competitions

Year	*Place*	*Year*	*Place*
1997	Nagoya	2003	Padua
1998	Paris	2004	Lisbon
1999	Stockholm	2005	Osaka
2000	Melbourne	2006	Bremen
2001	Seattle	2007	Atlanta
2002	Fukuoka	2008	Suzhou

RoboCup Categories

- Soccer
  - Humanoid league
  - Middle-sized league
  - Small-sized league
  - Four-legged league
  - Simulation league
- Rescue
  - Real robot league
  - Simulation league
- Junior
  - Soccer
  - Rescue
  - Dance
- RoboCup @home

Soccer Humanoid League

- Two-legged robots
- Ideal form for the ultimate aim?
- Category:

2-on-2 competition

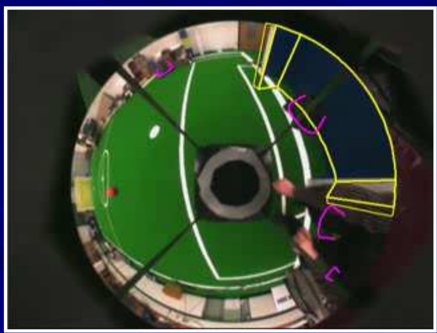
Penalty kick challenge

Soccer Middle-Sized League

- Maximum six players per team
- Fully autonomous mobile robots
- Wireless communication between players

Soccer Middle-Sized League

- Omni-directional move
- Omni-directional camera



Soccer Small-Sized League

- Five robots per team
- Global vision (with overhead camera)
- Remote software sending commands to robots

Soccer Four-Legged League

- Sony AIBO — Same robot condition
- Developing computer programs

RoboCup Soccer Simulation

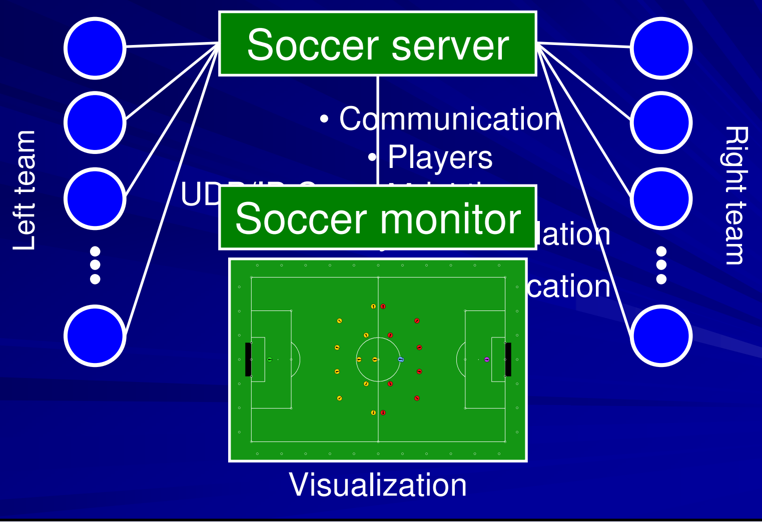
of

er

RoboCup Soccer Simulation

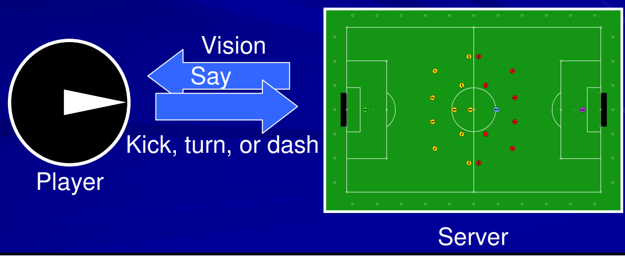
- Game format in RoboCup 2007 (Atlanta)
- Qualification (44 teams)
  – Top 3 teams in 2006 are automatically qualified
  – Top 8 teams in qualification round
  – 5 teams from scientific point of view (review of team description paper)
- In Atlanta (16 teams)
  – 8 teams to proceed to final tournament based on the results of two round-robin matches
  – Final tournament: double elimination

Soccer Server



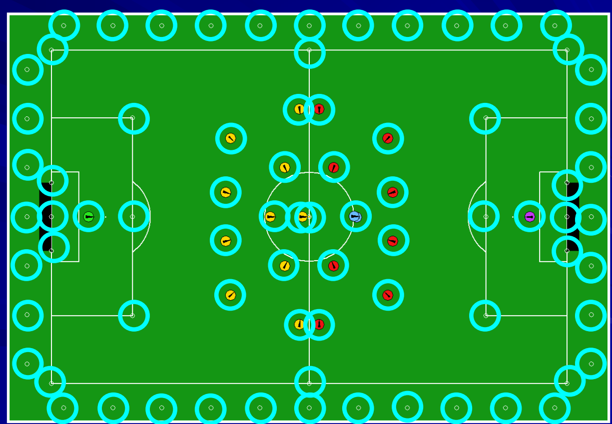
Player Model

- Communication with the server
  - Field information from the server
    - Vision
    - Aural
  - Action command to the server (one per cycle)
    - Kick
    - Turn
    - Dash
    - Say



Objects in Soccer Simulation

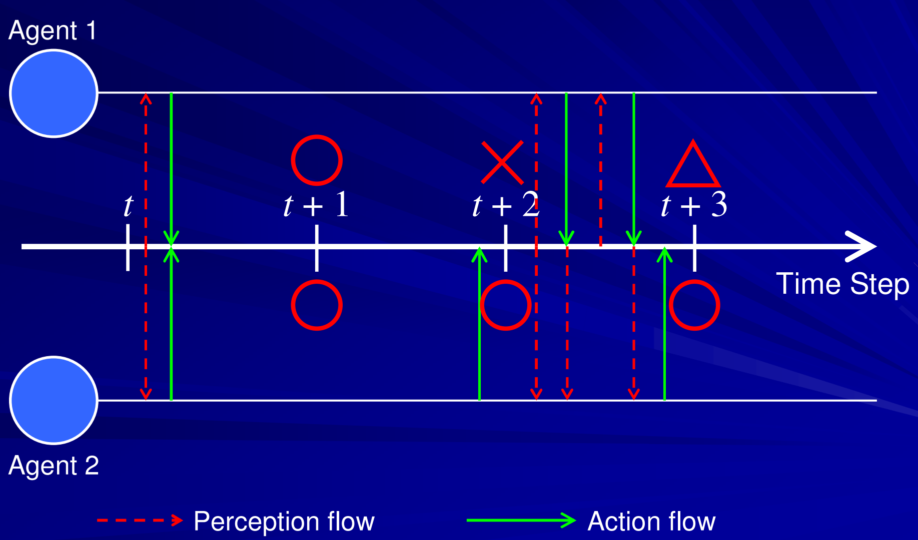
- Ball
  - Movement
- Players
  - Kick
  - Dash
  - Turn
  - Stamina
- Flag
  - Static



Simulation-Time Model

- Discrete-time system
  - Changing state for each time step
- Synchronous update of object position
  - One time step: 100 msec.
  - Each agent must send action within 100 msec.
- Asynchronous communication
  - Visual information is sent irregularly.
  - Action order can be sent anytime.

Simulation-Time Model



Movement Model (Position)

Position of an object at time step t : $\mathbf{p}^t$

Velocity of an object at time step t : $\mathbf{v}^t$

$$\boxed{\mathbf{u}^{t+1} = \mathbf{v}^t + \mathbf{a}^t}$$

Amount of movement

$$\boxed{\mathbf{p}^{t+1} = \mathbf{p}^t + \mathbf{u}^t}$$

Position at time step t+1

Movement Model (Velocity)

Position of an object at time step t : $\mathbf{p}^t$

Velocity of an object at time step t : $\mathbf{v}^t$

The velocity is updated after the position is updated.

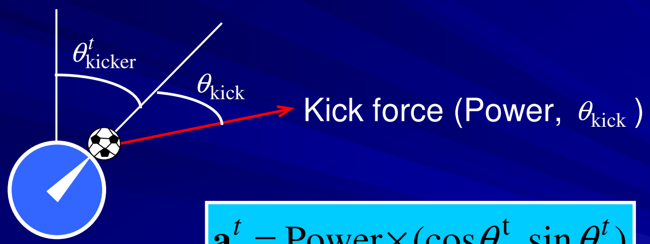
$$\boxed{\mathbf{v}^{t+1} = \mathrm{decay} \times \mathbf{u}^t}$$

Velocity at time step t+1

decay: A positive constant specified separately for players and a ball

Movement Model (Acceleration)

Acceleration for a ball is applied by *Kick* command.

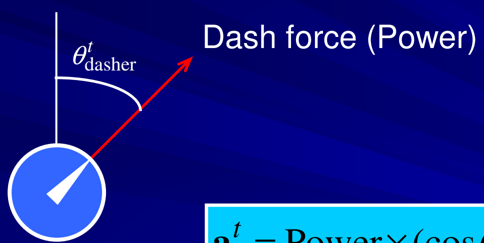


$$\mathbf{a}^t = \text{Power} \times (\cos\theta^t, \sin\theta^t)$$

$$\theta^t = \theta^t_{\text{kicker}} + \theta_{\text{kick}}$$

Movement Model (Acceleration)

Acceleration for a player is applied by *Dash* command.

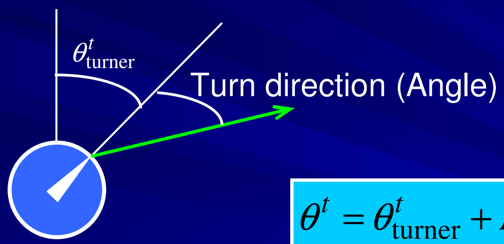


$$\mathbf{a}^t = \text{Power} \times (\cos\theta^t, \sin\theta^t)$$

$$\theta^t = \theta^t_{\text{dasher}}$$

Movement Model (Turn)

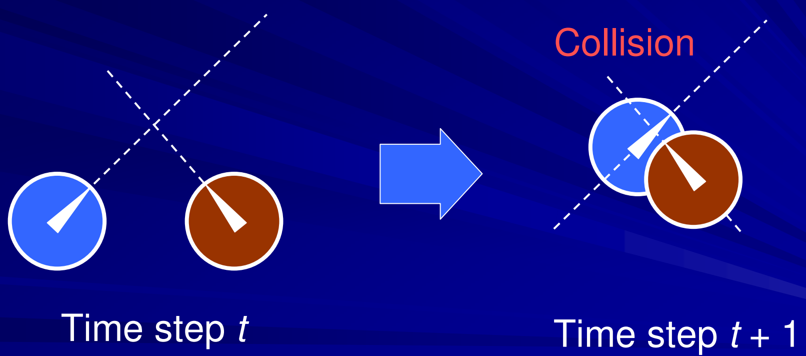
Turn command changes player's direction.



$$\boxed{\theta^t = \theta^t_{\text{turner}} + \text{Angle}}$$

Collision Model

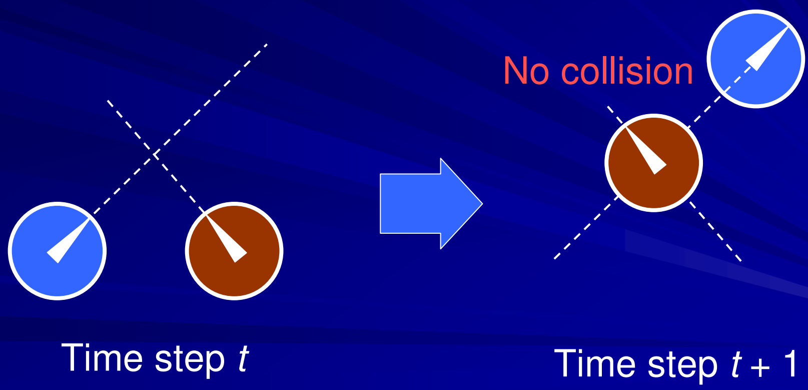
Two objects collide if they overlap after the movement update.



Two objects are randomly moved when collision occurs.

Collision Model

Two objects collide if they overlap after the movement update.



Noise Model for Acceleration

Uniformly distributed noise is added to the movement of all objects.

$$\mathbf{u}^{t+1} = \mathbf{v}^{t} + \mathbf{a}^{t} + \mathbf{r}$$

$$\mathbf{r} \in [-\mathrm{rand} \times |\, \mathbf{v}^{t} \,|, +\mathrm{rand} \times |\, \mathbf{v}^{t} \,|]$$

rand: A positive constant specified separately for players and a ball

Stamina Model

Reduction of stamina causes the limitation of maximum moving speed of a player.

stamina: The actual limit of dash power

effort: The efficiency of player movement

recovery: The recovery rate of stamina parameters

Stamina Model (Effect on Dash)

The effective dash power is determined by dash power, stamina, and effort.

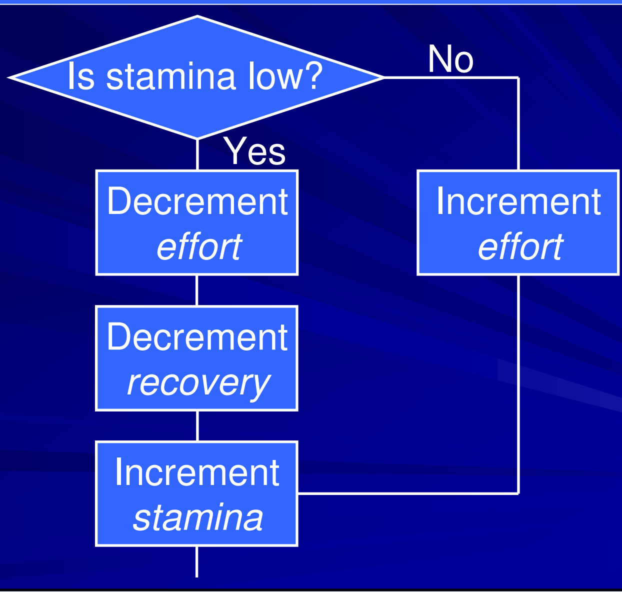
$$\text{Effective dash power} = \text{dash\_power\_rate} \times \min(\text{stamina}, \text{Power}) \times \text{effort}$$

dash_power_rate is a positive constant.

$$\text{stamina}^{\text{new}} = \text{stamina}^{\text{old}} - \text{Effective dash power}$$

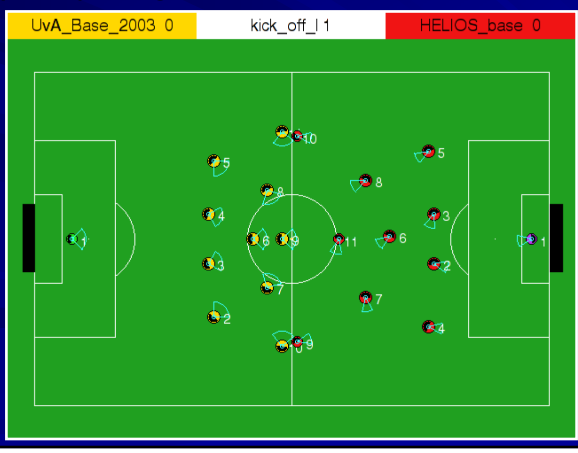
When a player is dashing reversely, the amount of decrement becomes twice the effective dash power.

Stamina Model (Stamina recovery)



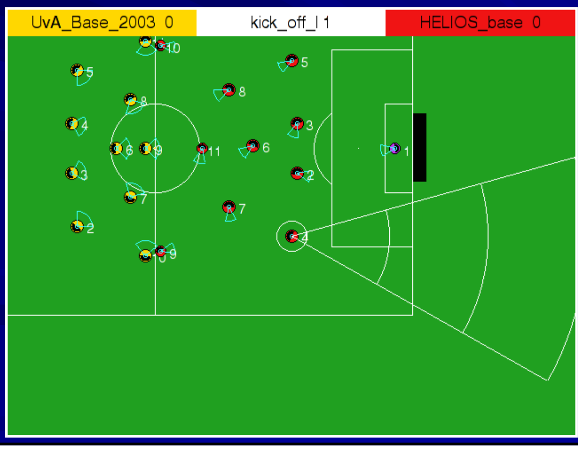
Vision Model

- View width: Narrow, normal, or wide
- View quality: High or low

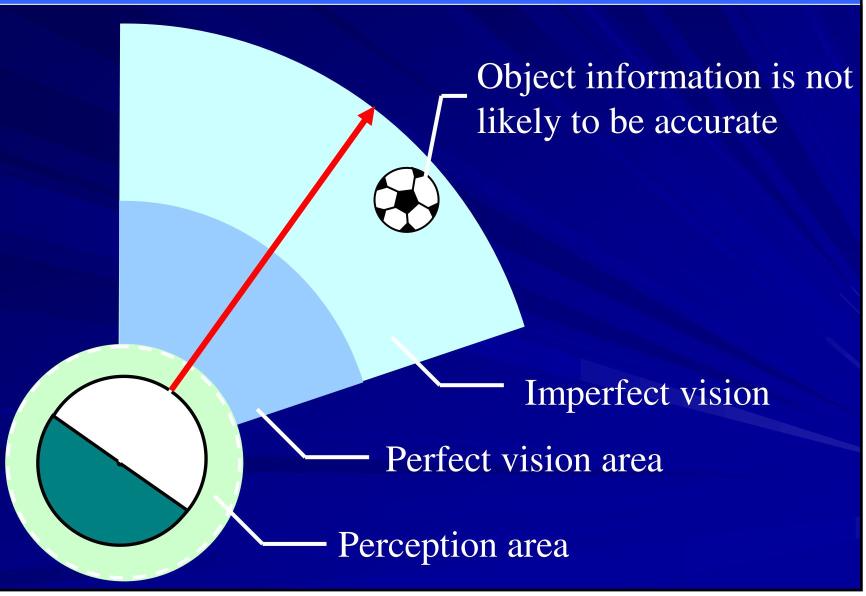


Vision Model

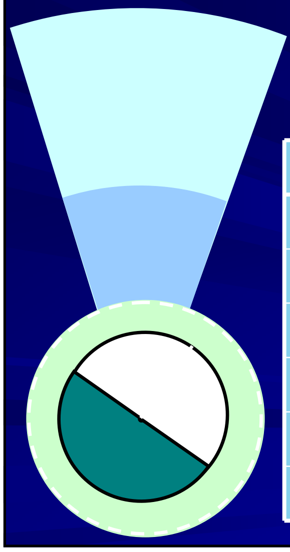
- View width: Narrow, normal, or wide
- View quality: High or low



Vision Model



Vision Model (View Mode)



Six view modes

- Quality: High, low
- Width: narrow, normal, wide

Qlt.	Width	Freq.(ms)	Information
High	Narrow	75	45°, Fine
High	Normal	150	90°, Fine
High	Wide	300	180°, Fine
Low	Narrow	37.5	45°, Rough
Low	Normal	75	90°, Rough
Low	Wide	150	180°, Rough

Precision of Visual Information

The soccer server sends uncertain visual information on objects to each player.

d: Actual distance, d': Visual information

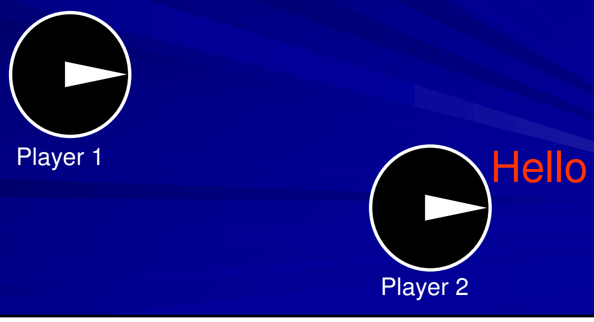
$$d' = \mathrm{Quantize}(\exp(\mathrm{Quantize}(\log(d), 0.1)), 0.1)$$ Player

$$d' = \mathrm{Quantize}(\exp(\mathrm{Quantize}(\log(d), 0.01)), 0.1)$$ Flag

$$\mathrm{Quantize}(V, Q) = \mathrm{rint}(V / Q) \times Q$$

Aural model

- Say command
  - Broadcast a message
  - Limited to ten bytes
  - Range: 50 meters

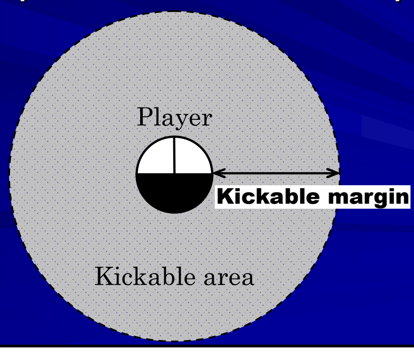


Kick/Dash/Turn Model

- Kick
  - Applies acceleration to the ball
  - Two parameters: Power and angle
- Dash
  - Player’s accelerations
  - Only forward direction
- Turn
  - Change the body angle of player

Kick Model

- Applies acceleration to the ball
- Valid when the ball is within *kickable_area*
- Two parameters: Power and angle
- Efficiency dependent on the ball position

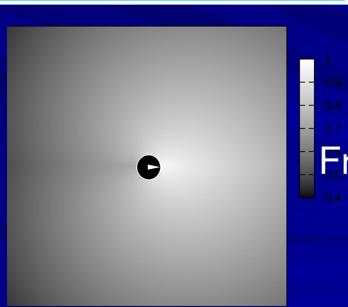


Kick Efficiency

$$\text{kick efficiency} = \text{kick\_power\_rate} \times \left(1.0 - 0.25 \cdot \frac{\text{dir\_dif}}{180} - 0.25 \cdot \frac{\text{dist\_ball}}{\text{kickable\_margin}}\right)$$

$$\text{effective kick power} = \text{power} \times \text{kick efficiency}$$

Rear



Front

Kick efficiency. Larger at lighter area

Dash Model

- Player's accelerations ([-100, 100])
- Only forward direction

$$\text{Effective dash power} = \text{dash\_power\_rate} \times \min(\text{stamina}, \text{Power}) \times \text{effort}$$

Turn Model

- Change the direction of player ([-180, 180])
- Less effective when speed is high

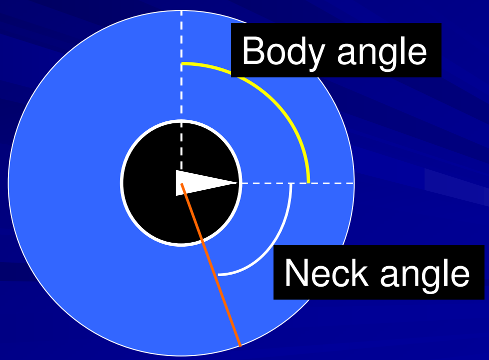
$$\text{Effective angle} = \text{angle}/(1.0 + \text{inertia\_moment} \times \text{player\_speed})$$

$$\Leftrightarrow$$

$$\text{angle} = \text{effective angle} \times (1.0 + \text{inertia\_moment} \times \text{player\_speed})$$

Neck Model

- Neck is independent part of the body ([-90, 90]).
- Two angles for a player: Body and neck

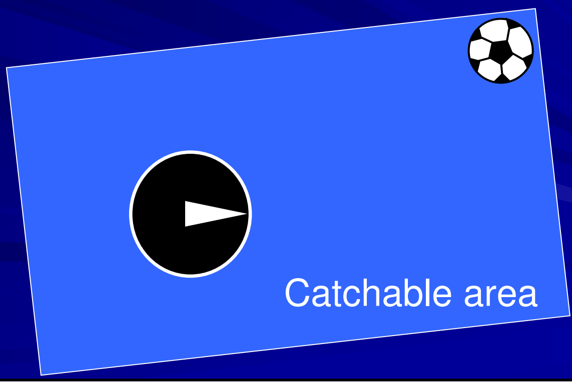


Tackle Model

- Forces the ball to accelerate along the body angle
- Increases the success probability if the ball is near the tackling player
- The player freezes for 10 time steps after tackle.

Catch Model

- Special action for goal keeper
- Catches the ball with the probability 100%
- One parameter: Relative angle to catch



Developing Soccer Agent

- UDP/IP connection
- Message parsing (S-expression)
- Time control
- Vision control
- Hetero selection
- Stamina management
- Decision making
- Formation
- Team coordination
- …

Base Teams

- UvA Trilearn Base (Netherland)
  - World Champion in 2003
  - C++ implementation
  - Simple strategy
  - http://staff.science.uva.nl/~jellekok/robocup/
- Dainamyte (Germany)
  - 9th place in 2007 and 2006
  - Java implementation
  - http://www.dainamite.de

Techniques for Developing Agents

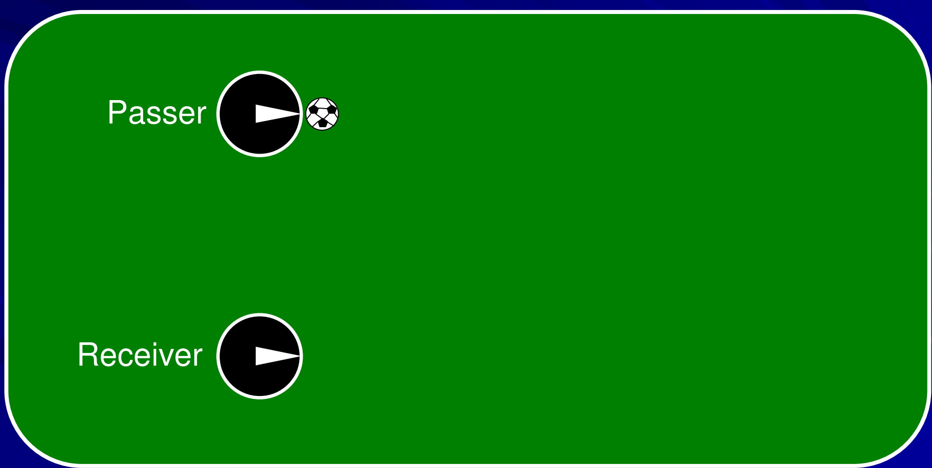
- Hand-coding
  - Embedding soccer skills into computer progs.
  - Depends on soccer knowledge
  - Depends on programming skill
- Self-learning
  - Reduce the necessity of domain knowledge
  - Letting players learn skills themselves
  - Computational Intelligence!

Computational Intelligence for RC

- Fuzzy systems for ball intercept
- Neural networks for mimicking dribble
- Evolutionary Computation for team strategy

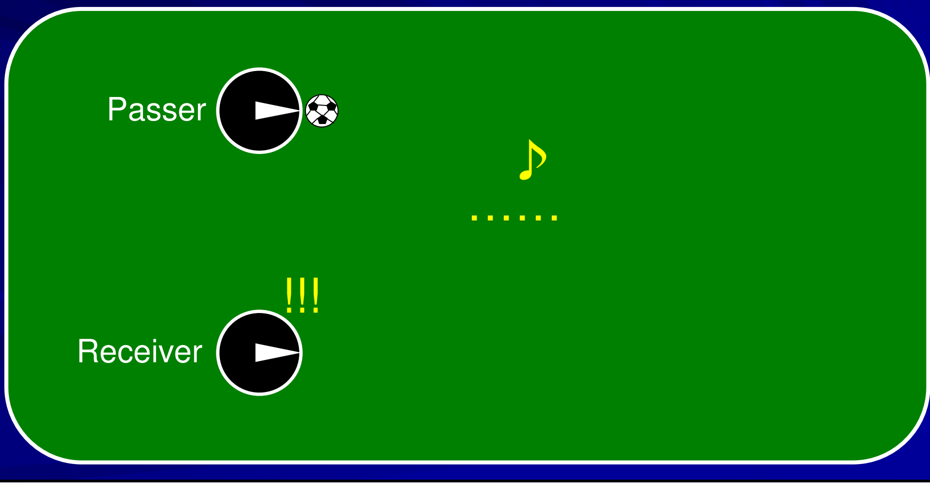
Fuzzy Systems for Ball Intercept

- Ball Intercept problem: To catch the ball



Fuzzy Systems for Ball Intercept

- Ball intercept problem: To catch the ball



Ball Intercept

- What's problem?

$$(v_x^{t+1}, v_y^{t+1}) = \underline{decay} \times \{ (v_x^t, v_y^t) + \boxed{(a_x^t, a_y^t)} + \underline{noise} \}$$

[Ball:0.94, Player:0.4]

Efficiency depends on stamina

Mathematical estimation does not help in the prediction of objects…..

Fuzzy If-Then Rules

R_i : If x_r is A_{i1} and y_r is A_{i2}
and v_{rx} is A_{i3} and v_{ry} is A_{i4}
then *turn* with w_i^{turn} and *dash* with w_i^{dash}
$i = 1, \ldots, N$

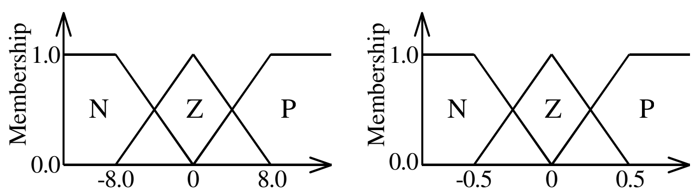
(x_r, y_r): Relative position of the ball

(v_{rx}, v_{ry}): Relative velocity of the ball

$(w_i^{turn}\ w_i^{dash})$: Real weights for action

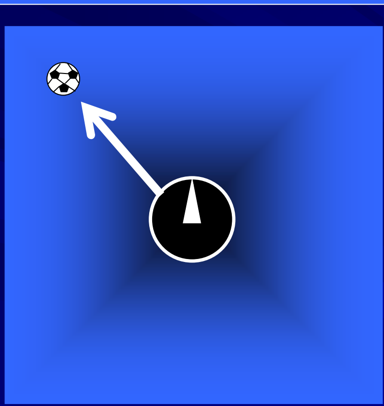
Fuzzy If-Then Rules

R_i : If x_r is A_{i1} and y_r is A_{i2}
and v_{rx} is A_{i3} and v_{ry} is A_{i4}
then *turn* with w_i^{turn} and *dash* with w_i^{dash}
$i = 1, \ldots, N$

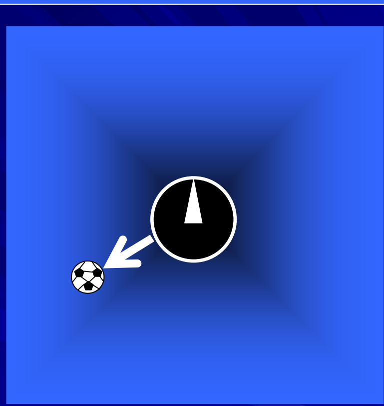


(a) Fuzzy partition for x_r and y_r (b) Fuzzy partition for v_{rx} and v_{ry}

Fuzzy Systems



Relative position



Relative velocity

The above information is used to determine whether the agent should dash or turn.

Adaptive Process

Fuzzy If-Then Rules

↓

Support Calculation

↓

Action Selection

↓

Updating Fuzzy If-Then Rules

Fuzzy Reinforcement Learning

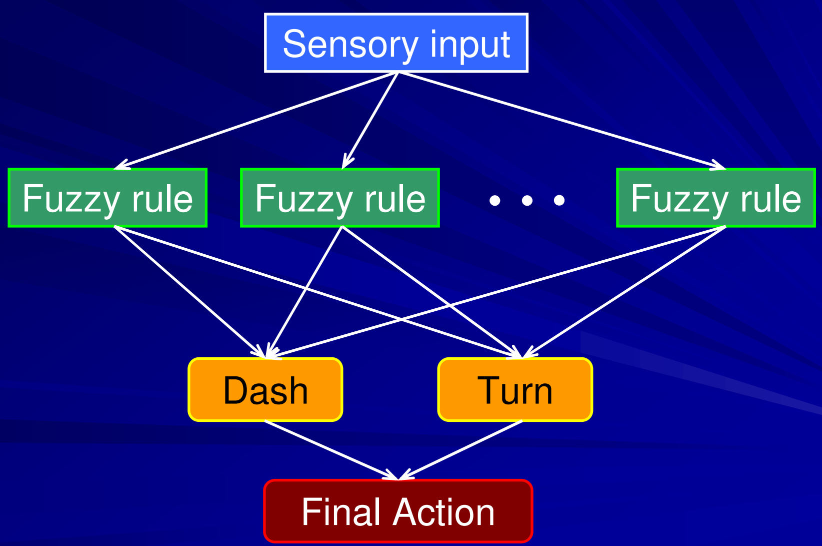
Support degree of action:

$$W_k = \frac{\sum_{i=1}^{81} w_{ik} \cdot \mu_i(\mathbf{s})}{\sum_{i=1}^{81} \mu_i(\mathbf{s})}, \quad k = turn, dash$$

Compatibility definition:

$$\mu_i(\mathbf{s}) = \mu_{i1}(x_r) \cdot \mu_{i2}(y_r) \cdot \mu_{i3}(v_{rx}) \cdot \mu_{i4}(v_{ry})$$

Fuzzy Reinforcement Learning



Fuzzy Reinforcement Learning

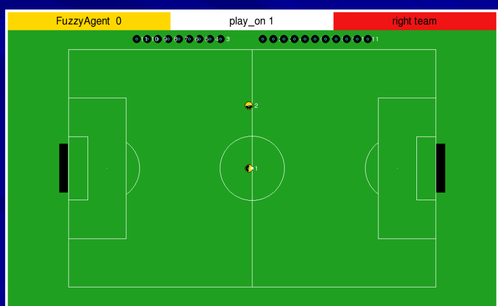
$$w_i^k := (1-\alpha'_{ik}) \cdot w_i^k + \alpha'_{ik} \cdot (r + \gamma \cdot W_{max})$$

$$\left(\begin{array}{l} \gamma : \text{Positive constants}, \quad r : \text{Reward} \\ W_{max} : \text{ Maximum value of } W^k \end{array} \right)$$

$$\alpha'_{ik} = \alpha \cdot \frac{\mu_j(\mathbf{s})}{\sum_{l=1}^{81} \mu_l(\mathbf{s})}$$

Demonstration (1):

- Fixed angle and fixed power
  - •Initial stage
  - •Intermediate stage
  - •Final stage
- Different angle



Demonstation (2):

- Random angle and fixed power
  - Initial stage
  - Intermediate stage
  - Final stage

Fuzzy System for Ball Intercept

- Adaptive system – On-line learning
- Learn how to move over time

- Issues to be addressed:
  - Tuning membership functions
  - Learning other behavior?

Hints for Applying EC for Ball Intercept

- Possible objectives
  - To minimize time steps to intercept ball
  - To maximize x-coordinate of intercept point
- Rule base-optimization
  - Antecedent fuzzy sets
  - Rule weights
  - Number of fuzzy rules
- Hybrid of on-line learning and EC
  - Memetic algorithm

Neural Networks for Mimicking

- Learning from observation

What would you do if you want to learn this trick?

Collect videos

↓

Watch them a number of times

↓

Try to mimic

Log File for Soccer Simulation

- Logs
  - *.rcg file (Binary format)
    - Position
    - Velocity
    - Stamina
  - *.rcl file (ASCII format)
    - Action sent to the server
    - Say message

Launch log player

Learning Procedure

Collect Log Files of Target Player

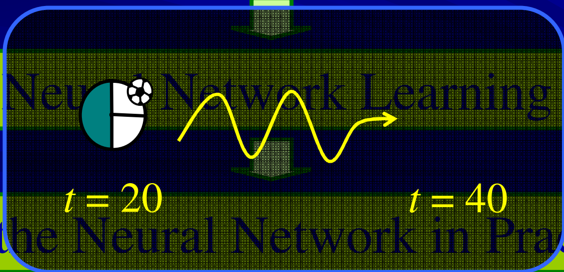
Use ... ice

Learning Procedure

Collect Log Files of Target Player

Extract Dribble Behavior

Neural Network Learning



Use the Neural Network in Practice

Learning Procedure

Collect Log Files of Target Player

Extract Dribble Behavior

Neural Network Learning

Use the Neural Network in Practice

Neural Network Implementation

- 13 input units, 30 hidden units
  - Body angle, and velocity of the target player
  - Position and velocity of ball
  - Relative position of three nearest opponent players
- Turn-neural network
  - Returns turn angle
- Kick-neural network
  - Returns kick angle and kick power
- Dash-neural network
  - Returns dash power

Learning Procedure

Collect Log Files of Target Player

↓

Extract Dribble Behavior

↓

Neural Network Learning

↓

Use the Neural Network in Practice

Experiments

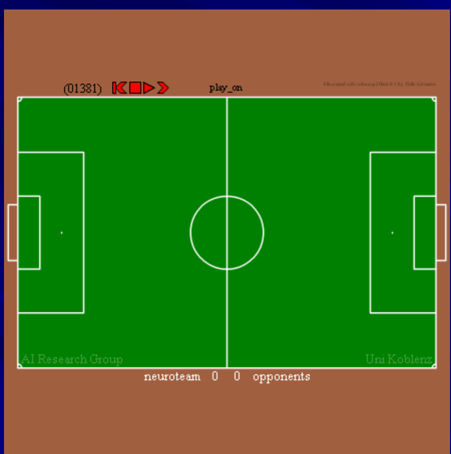
- Target team: STEP (Russia)
  - Winner of RoboCup 2004
  - Good dribble skill
- Collecting dribble information
  - Collect the games of STEP (72000 cycles)
  - Manual extraction of dribble intervals
- Training data set for neural networks
  - 478 patterns for turn
  - 6871 patterns for dash
  - 2359 patterns for kick

Demonstration: Before learning

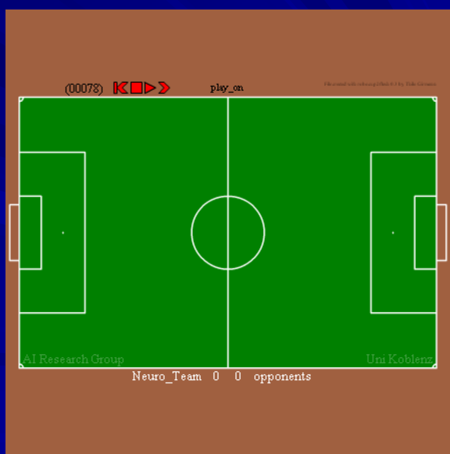
Acquired Dribble Skill

Comparison with STEP

STEP



Neural networks

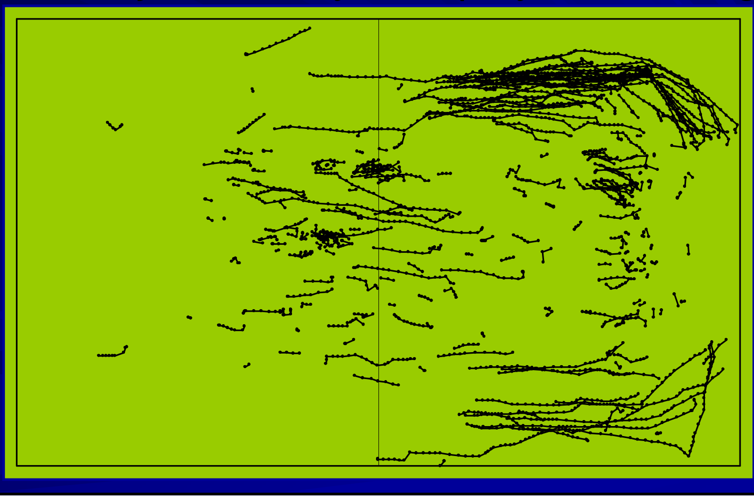


Another Experiments

- Target team: UvA Trilearn (Netherland)
  - Winner of RoboCup 2003
  - Well-balanced team
- Collecting dribble information
  - Collect 10 games of STEP (60000 cycles)
  - Automatic extraction of dribble intervals
- Criteria of dribble
  - Two succeeding kicks made by the same player

Automatically Extracted Dribble

Dribble trajectories by UvA players in one game



Two Neural Networks for Dribble

Neural network 1

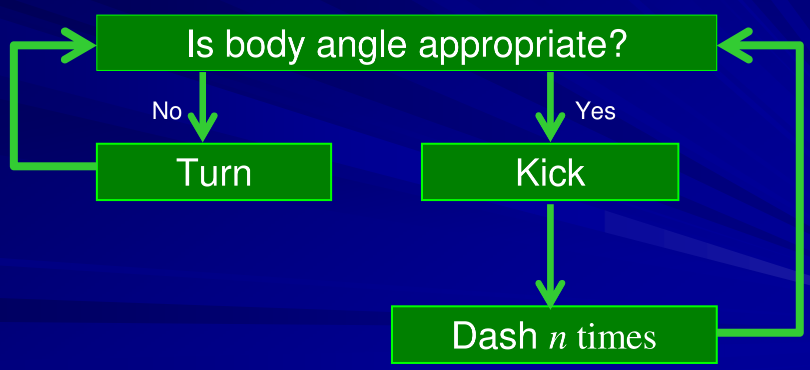
Ball position → Dribble direction

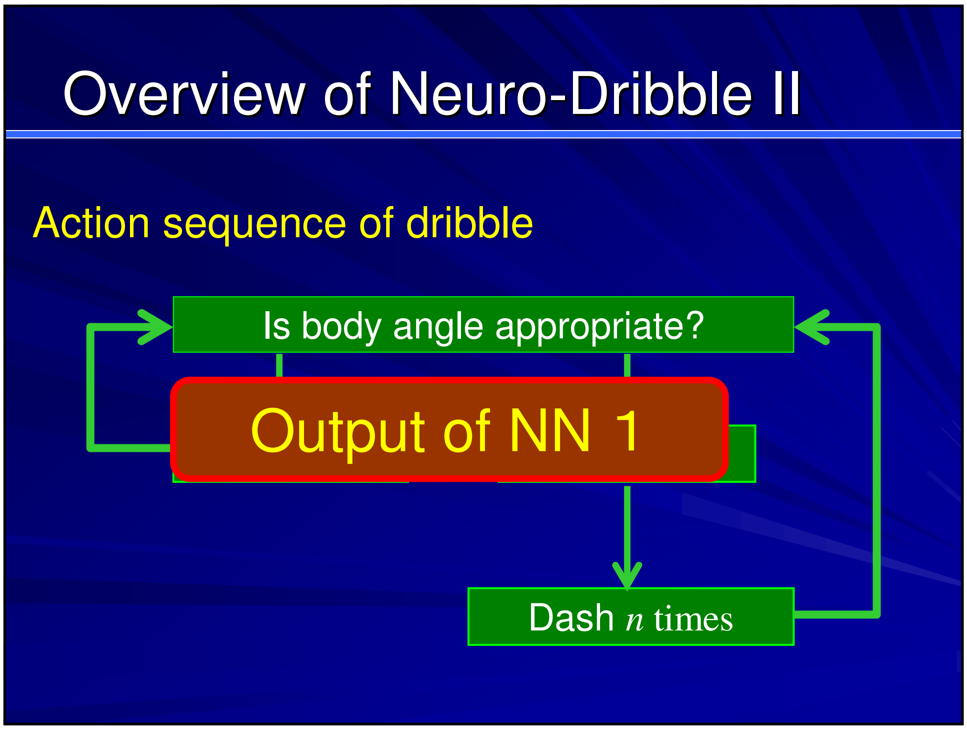
Neural network 2

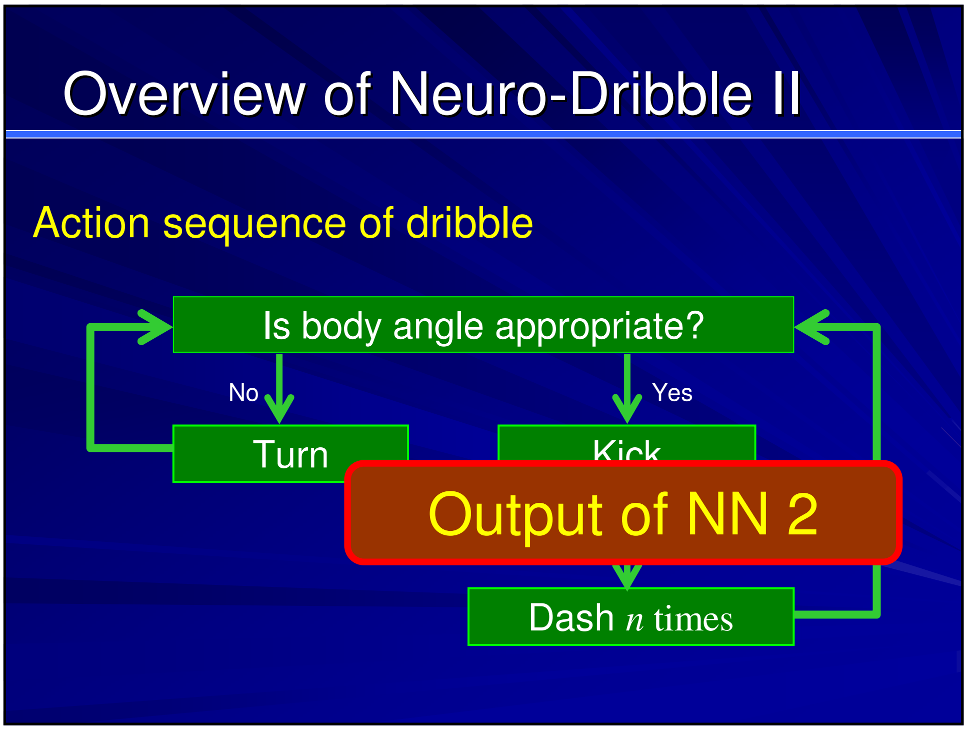
Ball position
Nearest opponent's position → Number of dashes

Overview of Neuro-Dribble II

Action sequence of dribble




Overview of Neuro-Dribble II
Action sequence of dribble
Is body angle appropriate?
Output of NN 1
Dash n times


Overview of Neuro-Dribble II
Action sequence of dribble
Is body angle appropriate?
No
Yes
Turn
Kick
Output of NN 2
Dash n times

Mimicking Dribble by NNs

- Quite promising
- Worked in real games
- Issues to be addressed:
  - Input selection
  - Recurrent structure of NNs
  - Human players as the target

Hints for Applying EC for Neuro-Dribble

- Possible Objectives
  - Speed
  - Some measure for good dribbling direction
- Neural Network-Optimization
  - Standard or Recurrent
  - Input Selection
  - With or without back-propagation when standard NNs are used
- Need to Overcoming the Original
  - Optimization against the original

Evolutionary Computation

- Low-level behavior
  - Ball intercept
  - Dribble
  - Shot
  - etc.
- High-level behavior
  - Team strategy

Structure of Team Strategy

Team strategy

Offensive strategy

Defensive strategy

Defensive Strategy (Without Ball)

- If the agent is closest to the ball in its team, the agent moves toward the ball.
- Otherwise, the agent tries to keep its relative position with respect to its home position and the ball position.

Structure of Team Strategy

Team strategy

Offensive strategy

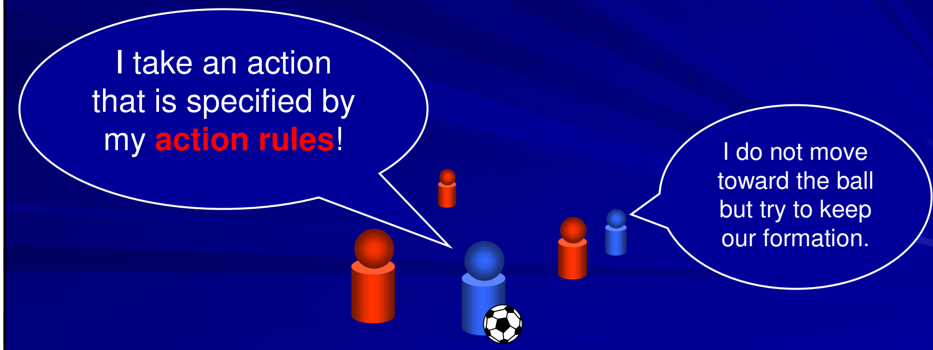
Defensive strategy

Objective

Improvement of the offensive performance

Offensive Strategy (With Ball)

- The agent chooses an action such as ***pass***, ***dribble***, ***shot*** using its action rules.



Action Rule

R_j : If the agent is in Area A_j

and the nearest opponent is B_j

then the action is C_j, $\quad j=1,\ldots,N$

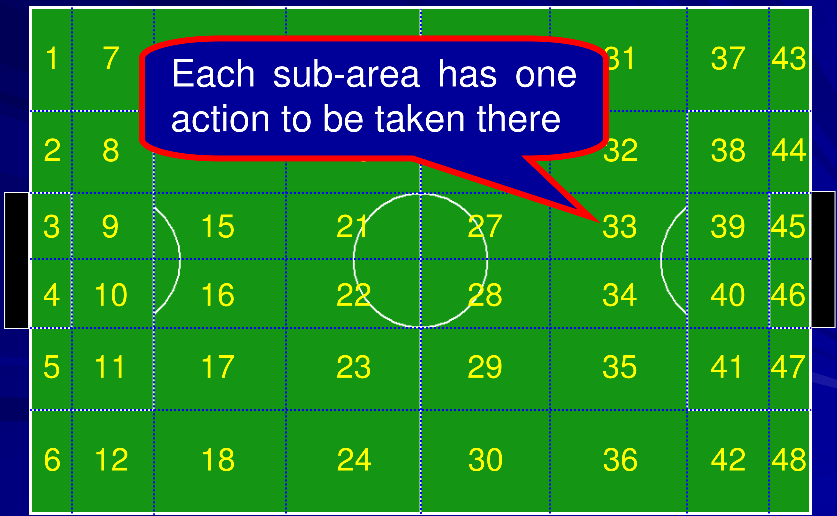
R_j : Rule label

A_j : Antecedent area label

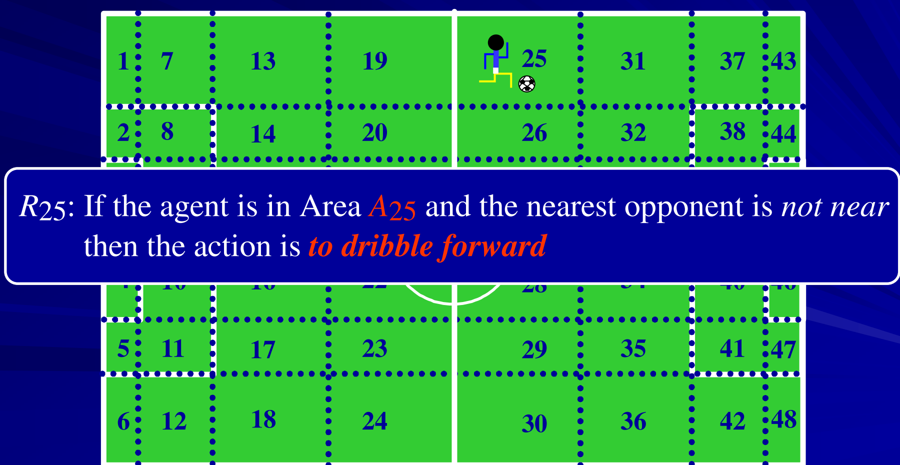
B_j : "*near*" or "*not near*"

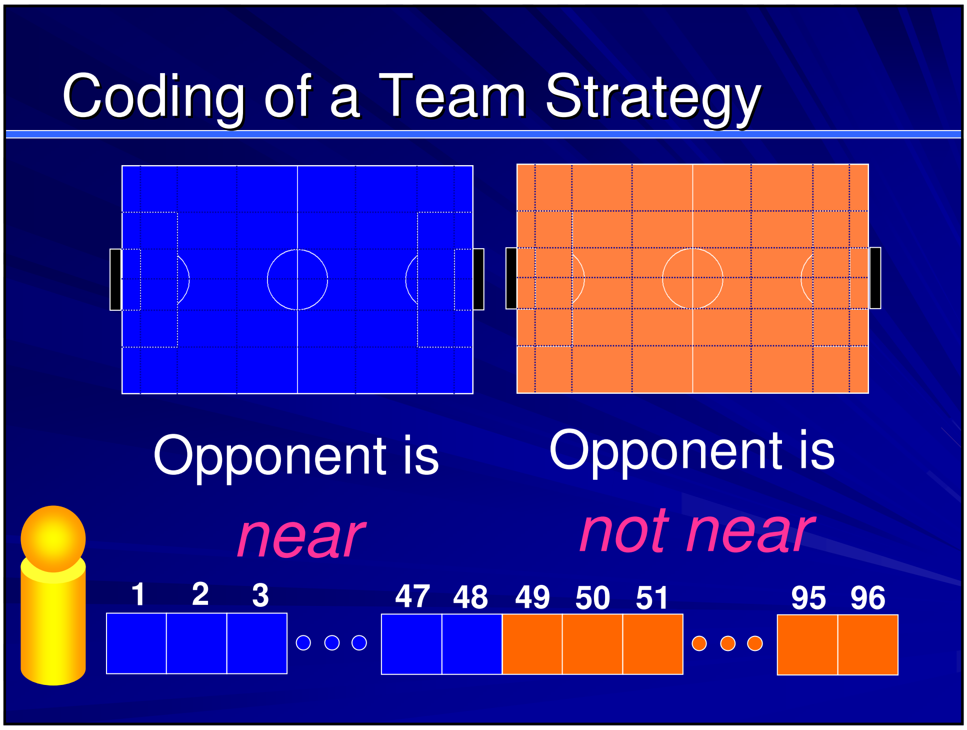
C_j : Consequent action

Partition of Soccer Field

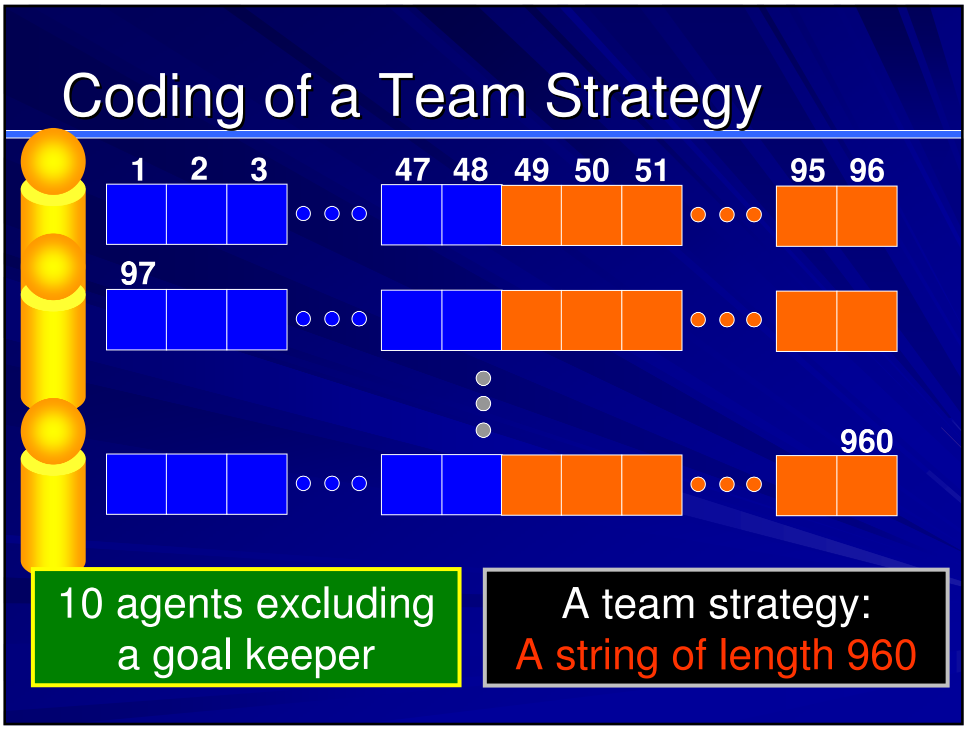


Action Selection





Coding of a Team Strategy
Opponent is
near
Opponent is
not near
1 2 3
47 48 49 50 51
95 96

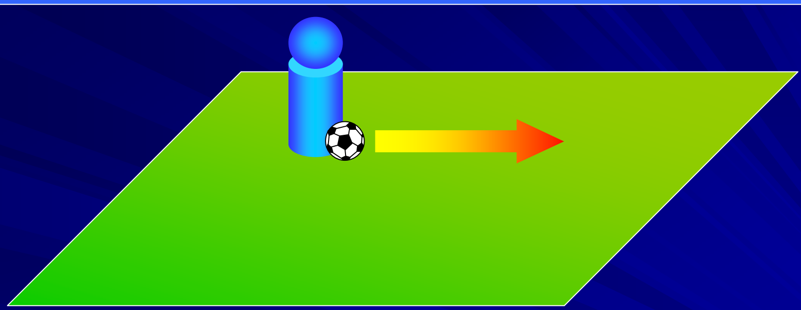


Coding of a Team Strategy
1 2 3
47 48 49 50 51
95 96
97
960
10 agents excluding a goal keeper
A team strategy:
A string of length 960

Coding of a Team Strategy

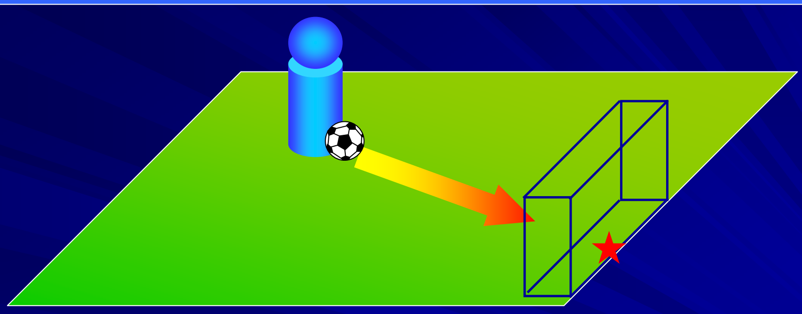
Dribble	6 types
Pass	2 types
Clearance	3 types
Centering	1 type
Shot	1 type
	13 choices

Example of Basic Action (*Dribble*)



Action 1:
Dribble toward the opponent side. The direction is parallel to the side lines.

Example of Basic Action (*Dribble*)



Action 2:
Dribble toward the center of the opponent goal.

Procedure of Match

- Matches against the common opponent team

… V.S.

Individuals Opponent team

- Performance evaluation through match results
  - 1st criterion: Goals for
    - 5-3
    - 4-1
    - is better!
  - 2nd criterion: Goals against
    - 5-3
    - 5-1
    - is better!

Evolutionary Operations

Parents

Offspring generated from the parents

Selection

Binary tournament selection with replacement

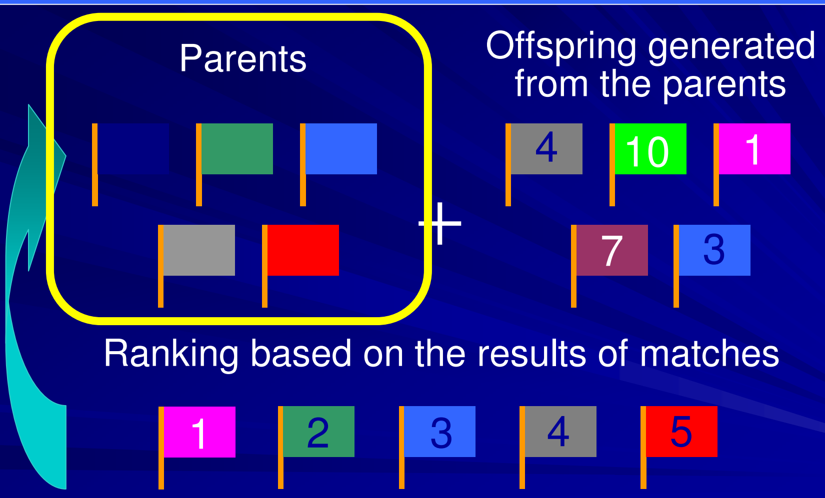
Crossover

One-point crossover

Mutation

Structured Mutation

Evolutionary Operations



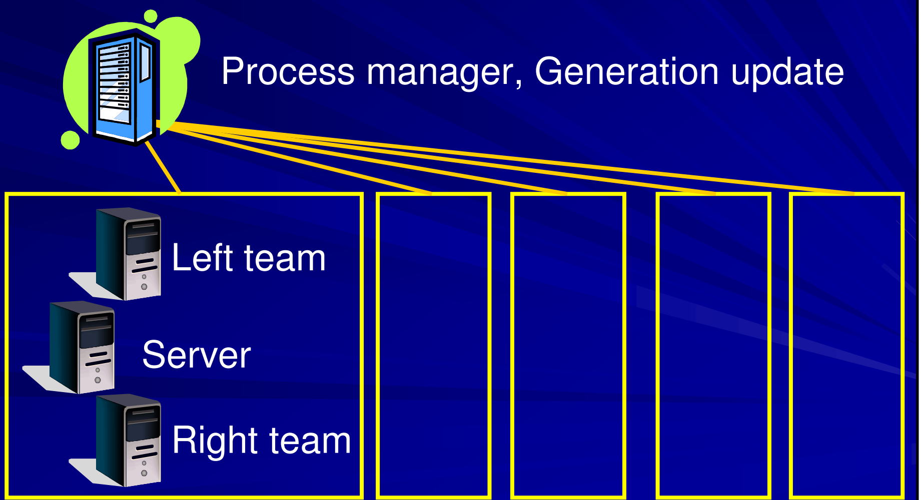
Best five teams → Next generation

Experimental Setting

Population size	5
The number of offspring strategies	5
The probability of mutation for each integer value	1/96
Generation of initial team strategies	Hand-coding:1 Random:4

Experimental using Cluster

A cluster system with 16 PCs are used for experiments.



Simulation Results

Generation	Win	Loss	Draw	Goals f.	Goals agst.
0	**1**	9	0	**3**	28
100	**2**	4	4	**11**	15
200	**3**	5	2	**11**	12
300	**7**	2	1	**15**	10

See the evolved team

EC for Obtaining Team Strategies

- Take enormous time (10 mins. per game)
- Rough fitness, but nice strategies
- Issues to be addressed:
  - – Unstable game results
  - – Coding scheme
  - – Adaptive candidate action
  - – etc.

EC for Obtaining Team Strategies

- Possible Objectives
  - Goal difference
  - "Goodness" of individual plays during game
- Speeding-Up EC
  - More computers
  - Approximation of fitness without actual evaluation
- Subjective Evaluation
  - Interactive EC

Team OPU_hana

Since Spring Competition'02

- History

JapanOpen'03: Top 8

World Competition'03: Second round

JapanOpen'04: Second round

World Competition'04: Second round

World Competition'05: 3D league

World Competition'06: 3D league

Team OPU_hana (continued)

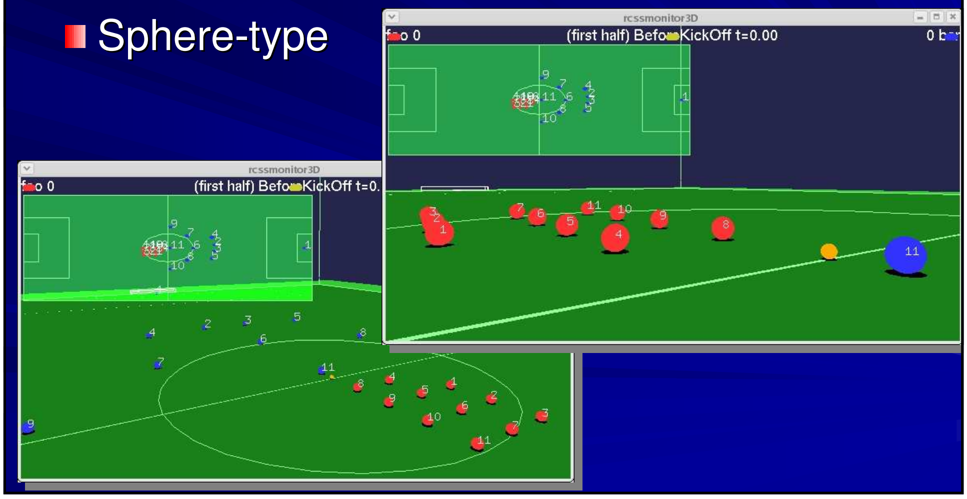
JapanOpen'07: Runner-up

World Competition'07: Top 4

Year	Method		
	Fuzzy	NeuralNet	EC
2002	○		
2003	○		○
2004			○
2005		○	○
2006		○	○
2007		○	

Future Direction of Soccer Simulation

- Development of 3D simulator
- Sphere-type



Future Direction of Soccer Simulation

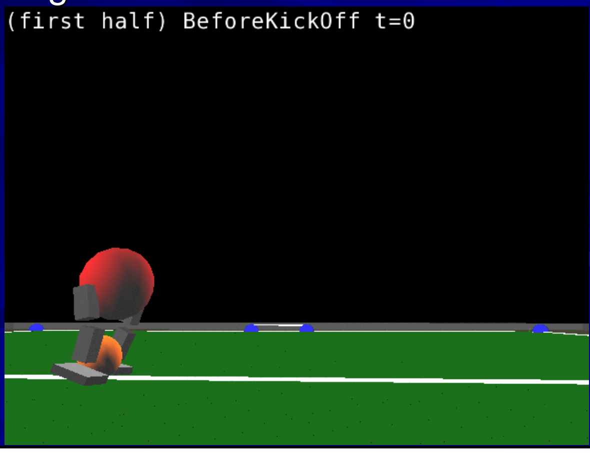
- Sphere agent → Legged agent

Future Direction of Soccer Simulation

- Sphere agent to legged agent
  - Six degrees of freedom

Future Direction of Soccer Simulation

- Sphere agent to legged agent
  - Six degrees of freedom



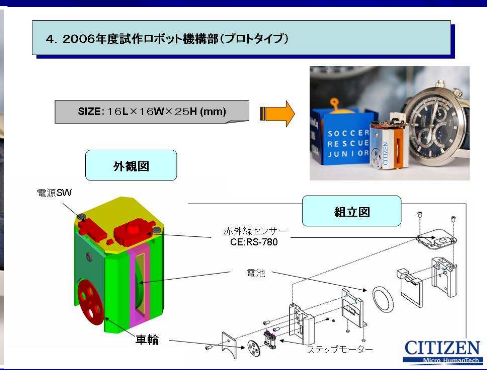
Future Direction of Soccer Simulation

- Humanoid simulation since 2007
  - Closer to real humanoid league
  - Complex to implement
  - Necessary to use control theory

Check the humanoid simulation

Future Direction of Soccer Simulation

- Physical visualization League
- Bridging the gap between simulation and real robot leagues

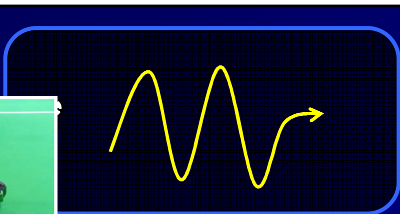


Future Direction of Soccer Simulation

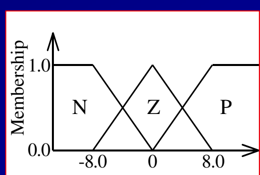
- Physical visualization L
- Bridging the gap betw
- real robot leagues

Conclusions

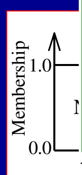
- RoboCup Soccer Simulation
  - Attractive
  - Good for research and education
- Computational Intelligence techniques
  - Fuzzy system for ball intercept
  - Neural networks for mimicking dribble
  - Evolutionary computation for team strategies
- Future directions
  - 3D humanoid robots, PV robot



Thank you!



(a) Fuzzy partition for x_r and y_r



(b) Fuzzy partition for v_{rx} and v_{ry}