Optimizing Interdependent Skills for Simulated 3D Humanoid Robot Soccer

Daniel Urieli, Patrick MacAlpine, Shivaram Kalyanakrishnan, Yinon Bentor, Peter Stone

UT Austin Villa

The University of Texas at Austin

Goal

Creating and integrating a set of motion skills for a 3D simulated robot soccer player

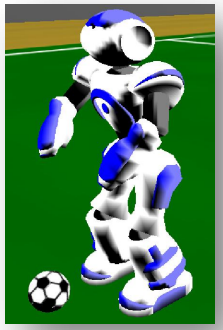
Background

- Simspark simulation
- Based on ODE engine
- Robot model: Aldebaran's Nao
- Message-based interaction with simulator

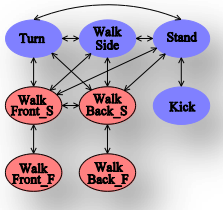
- 22 degrees of freedom
- Communication between agents – 20 bytes messages
- A robot is operated by joint torques
- We wrapped it with a PID controller

Contributions

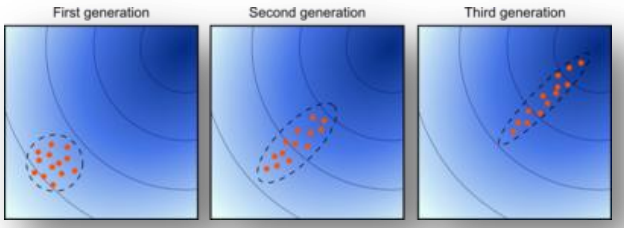
- A skill learning architecture for a humanoid robot soccer agent
  - Fully deployed in Robocup 2010
  - Learning rather than hand-coding more than 100 parameters
  - A significant building block in our agent, which is competitive with top-8 agents of Robocup 2010



- Sheds light on designing fitness functions for constraining an evolutionary learning process

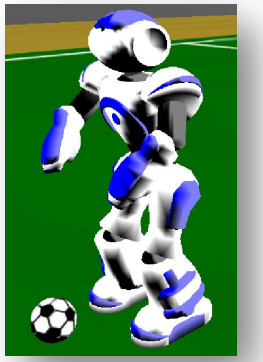


- A new successful application of the CMA-ES algorithm



The Need for A Learning Architecture

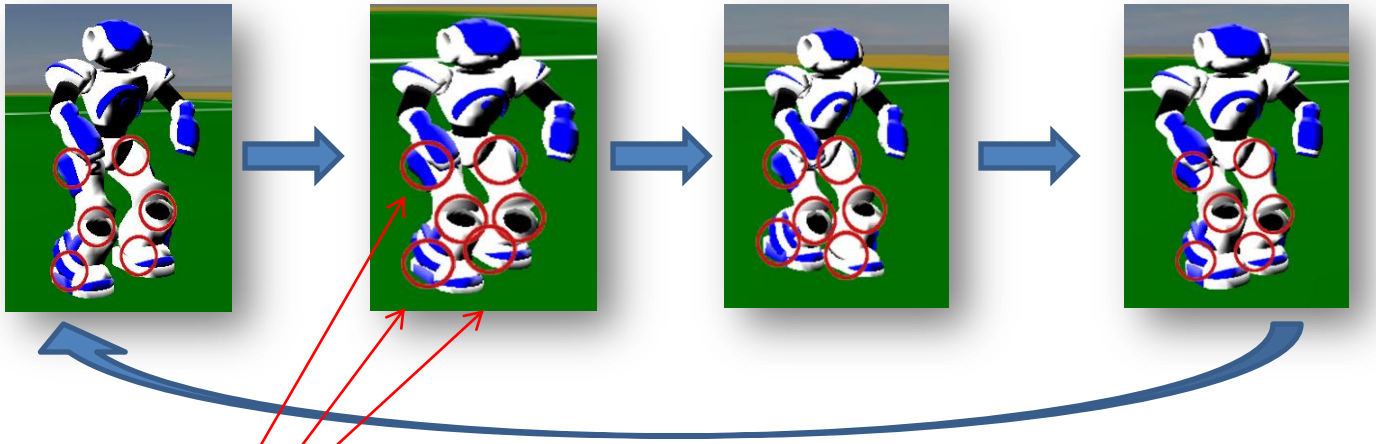
- Skills needed by a soccer playing robot:
  - Walk-front
  - Walk-back
  - Walk-diagonally
  - Walk-sideways
  - Turn
  - Kick
  - Goalie-dive
  - More…



- Coding each skill by hand might be tedious and sub-optimal
- On top of it, a skill design need to account for cooperation with other skills
  - A robot running full speed forwards need to be able to stop and turn without falling….
- Calls for a skill learning architecture

A Framework for Optimization through Learning

- Open loop joints control
- Repeatedly execute 4 control frames



Each frame specifies direct joint angles

Skills Description Language

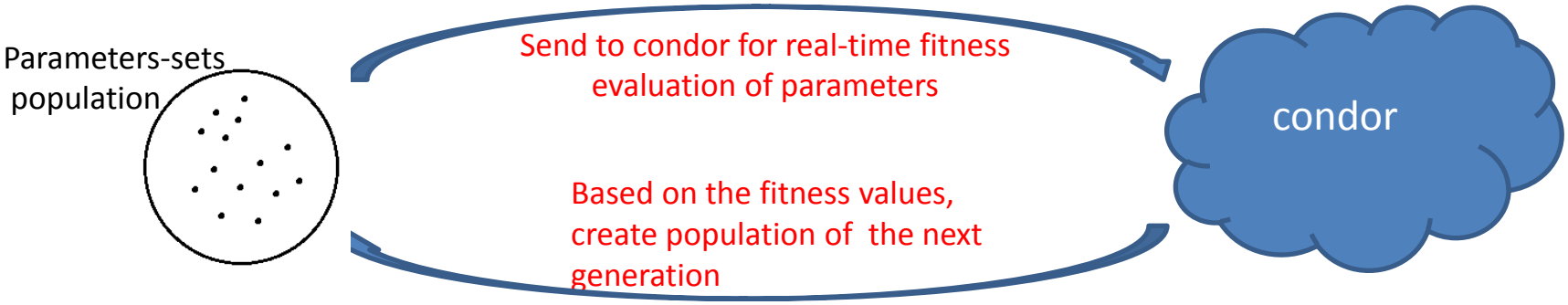
```
SKILL WALK_FRONT

KEYFRAME 1
reset ARM_LEFT ARM_RIGHT …
setTarget JOINT1 $jointvalue1 JOINT2 $jointvalue2
setTarget JOINT3 4.3 JOINT4 52.5
...
wait 0.08

KEYFRAME 2
...
```

Running Massive Amounts of Jobs in Parallel

- Our framework uniformly implements several evolutionary algorithms for parameters learning
- Evaluations are done in parallel using **Condor (*www.cs.wisc.edu/condor*)** - an open source software for parallel computing
- Repeatedly:

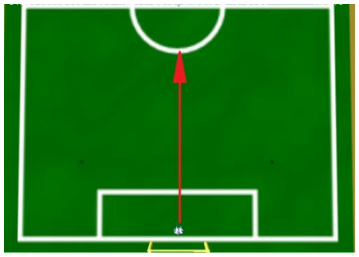


- A complete learning experiment contains 15,000-50,000 runs
  - For instance, 100 generations x 100 population x 5 averaging runs
  - Using condor, we run 100 simulations in parallel, 25 seconds per simulation
  - Wall clock time is 5-7 hours, for a total CPU time of ~350 hours

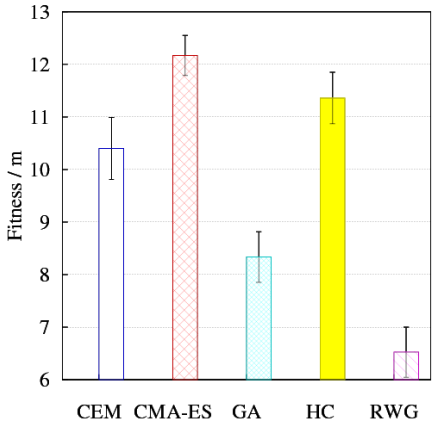
Optimizing individual skills

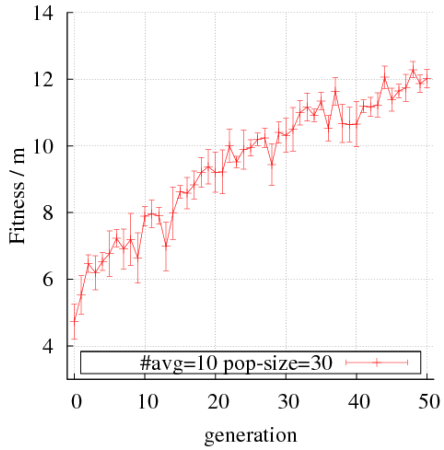
- **Goal: optimize the set of joint angles for maximum speed**

- A fitness of a set of joint angles:
  The agent's displacement in the desired direction
  - Inherently accounts for falls and non-straight walks
  - Measured over 15 seconds



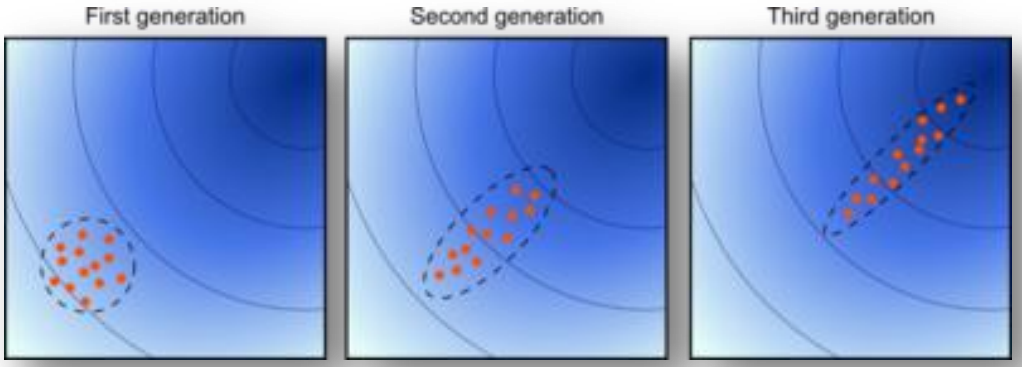
- Extensively compared several learning algorithms:
  - Hill-Climbing, Cross-Entropy Method, Genetic Algorithm and CMA-ES





CMA-ES learning curve

CMA-ES



- A stochastic, derivative-free, evolutionary numerical optimization method for non-linear or non-convex problems
- Each generation, candidates are sampled from a multidimensional Gaussian, and evaluated for their fitness
- Two main principles for parameter adaptation:
  - Mean maximizes the likelihood of previously successful candidates, Covariance maximizes the likelihood of previously successful search steps (Natural Gradient Decent)
  - Evolution paths are recorded and used as an information source

Found out to be extremely effective in our domain

Results – Individual Skills

Skill	Statistic	Performance
WalkFront	Speed	1.07(.00) m/s
WalkBack	Speed	1.03(.00) m/s
WalkSide	Speed	.62(.01) m/s
Turn	Angular speed	112.03(.24) °/s
Kick	Ball displacement	5.09(.07) m

Front Walk


SimSpark
0 UTAustinVilla (1st half) BeforeKickOff t=0.00
<Right> 0

Back Walk

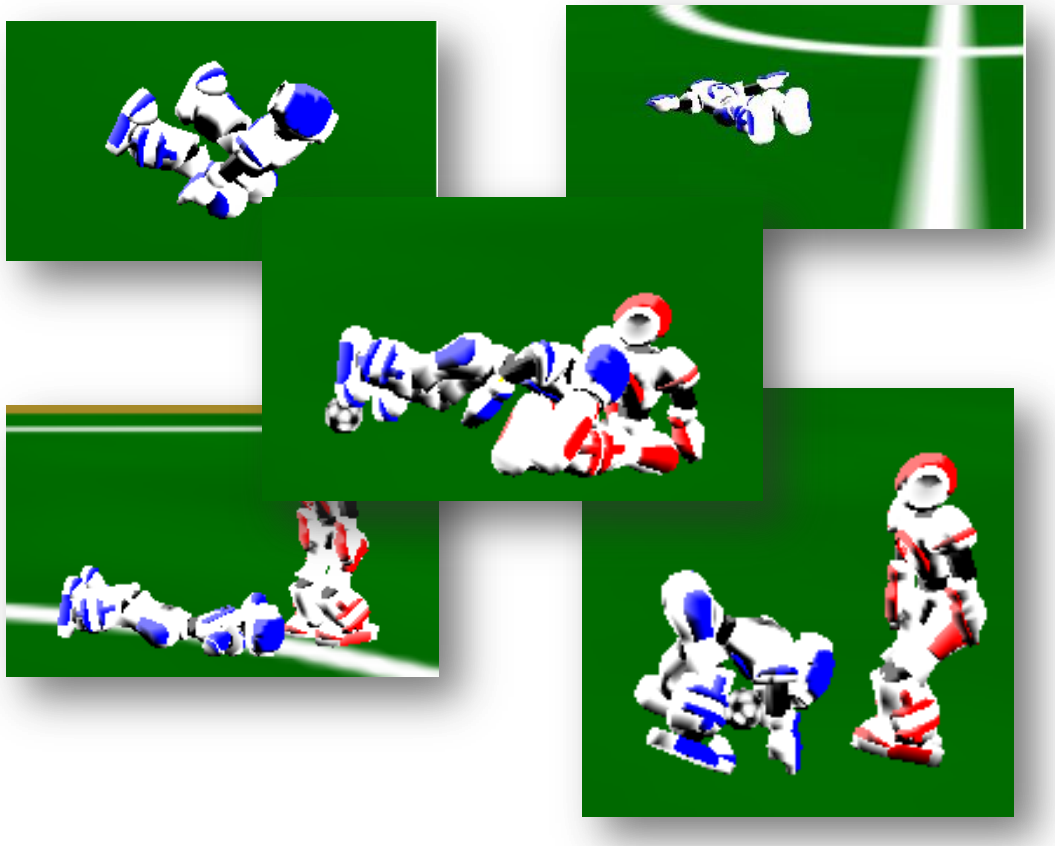

SimSpark
0 UTAustinVilla (1st half) BeforeKickOff t=0.00
<Right> 0

Kick


SimSpark
0 UTAustinVilla (1st half) BeforeKickOff t=0.00
<Right> 0

Optimizing Sequences of Skills

- Problem: fast locomotion skills, when integrated directly into the robot, result in frequent falls.



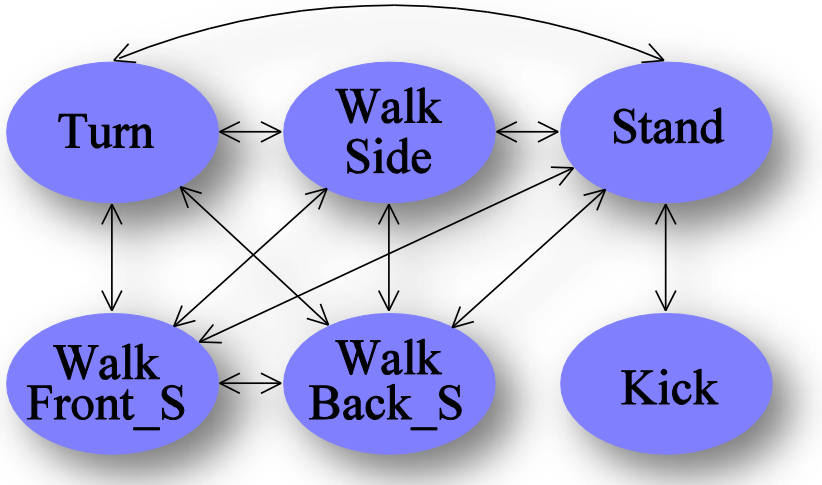
Optimizing Sequences of Skills

- Problem: fast locomotion skills, when integrated directly into the robot result in frequent falls.
- An example skill execution log (32ms decision cycle):

> WalkFront, WalkFront, Turn(R), Turn(R), Turn(R), WalkFront,
> WalkFront, WalkFront, Turn(L), Turn(L), WalkBack, ...

⟹ Skills are interdependent: Learn them together

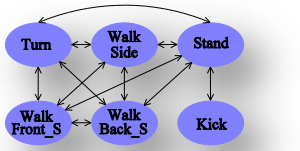
- Skills dependencies graph:



Idea 1: Optimize skills in conjunction

- Want both speed and stability under these transitions:

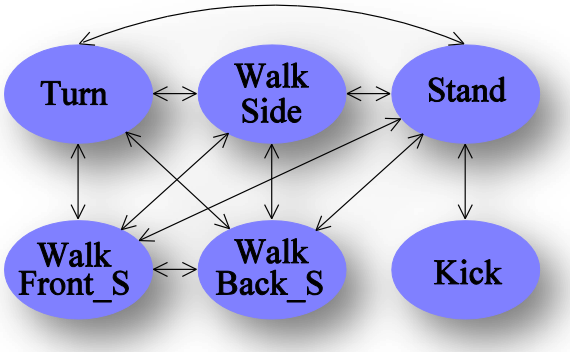
WalkFront, WalkFront, Turn(R), Turn(R), Turn(R), WalkFront,
WalkFront, WalkFront, Turn(L), Turn(L), WalkBack, ...



- Change the fitness evaluation method:
  - Evaluation method should include all skill transitions
  - But still reflect how good the currently-learned skill is
- An ideal fitness evaluation: Full Game results
  - But too noisy
- An effective alternative:
  - The time-to-score on an empty field
  - No noise caused by other players
  - Robot moves in a realistic scenario of skill transitions
  - Evaluated based on its ultimate objective

A Problem

- So far, optimized under these constraints

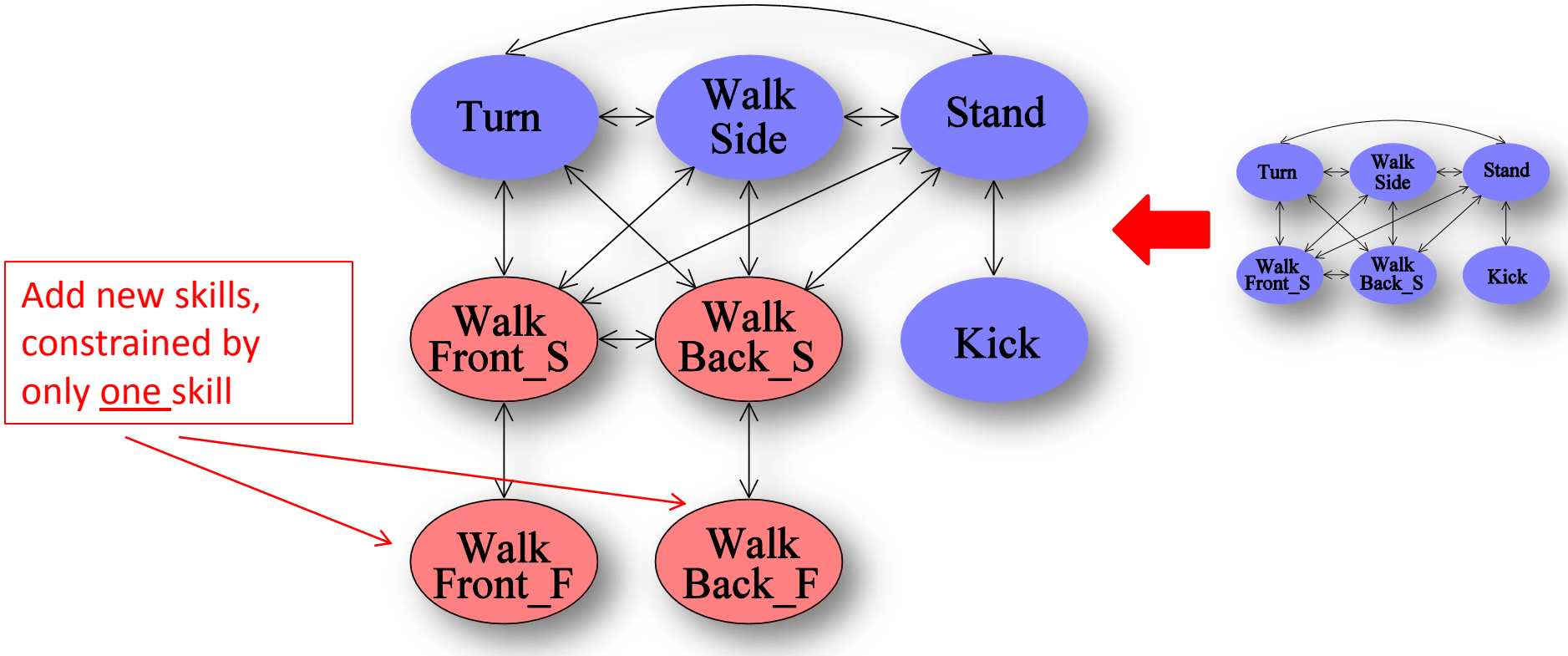


- The need to transition smoothly from every skill to every skill limits our max-speed

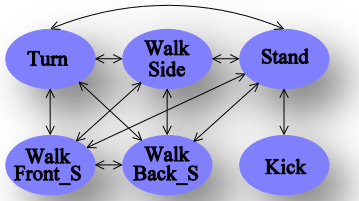
- Can we relax some constraints, thus achieving faster speeds?

Idea 2: Skill Decoupling

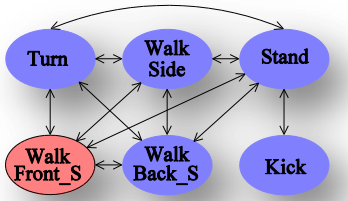
- It turns out we can further optimize speed, by adding additional, less-constrained skills.



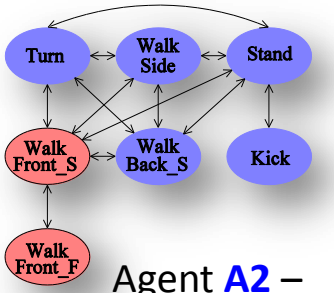
Putting it all together



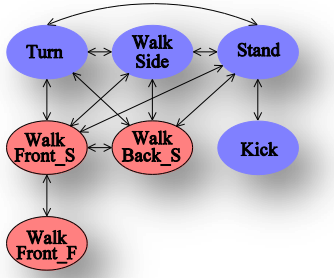
Agent **A0** – initial seed



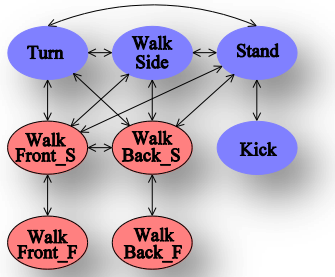
Agent **A1** – WalkFront_S optimized



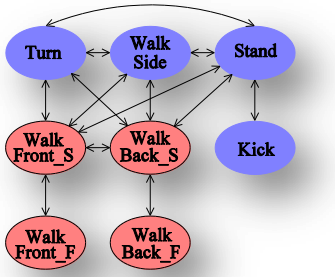
Agent **A2** – WalkFront_F optimized



Agent **A3** – WalkBack_S optimized



Agent **A4** – WalkBack_F optimized



Agent **A5** – Decision thresholds tuned

A0 vs. A5


SimSpark
0 A0 (1st half) BeforeKickOff t=0.00 <Right> 0

A0


SimSpark
0 UTAustinVilla (1st half) BeforeKickOff t=0.00 <Right> 0

A5

Results – Agents Improvements

TABLE II

GAME RESULTS BETWEEN AGENTS A0 THROUGH A5. ENTRIES SHOW THE GOAL DIFFERENCE (ROW – COLUMN) FROM 10 MINUTE GAMES.

	A0	A1	A2	A3	A4
A5	2.11(.10)	.77(.10)	.70(.10)	.58(.09)	.48(.08)
A4	1.66(.10)	.46(.08)	.15(.07)	.03(.07)	
A3	1.67(.10)	.28(.08)	.01(.08)		
A2	1.33(.10)	.20(.07)			
A1	1.23(.10)				

Full 6x6 game results

Results – Time-To-Score Measure

Agent	Time-To-Score/s
Apollo3d	31.08 (1.46)
A5	**34.49 (0.89)**
RoboCanes	36.18 (1.40)
NaoTH	36.75 (1.63)
UTAustinVilla	37.20 (0.89)
FCPortugal	47.54 (1.94)
SEURedSun	52.11 (2.49)
A0	**63.52 (1.05)**
Little Green Bats	71.02 (1.96)
FutK	77.89 (4.19)
BeeStanbul	98.56 (3.63)
Nexus3D	152.76 (5.15)
RoboPub	291.86 (1.17)
NomoFC	295.48 (1.32)
Bahia3D	300.01 (0.00)
Alzahra	300.01 (0.00)

Results – Full Games

Rank	Team	A0	A5
1	Apollo3d	4.29 (.17)	1.88 (.13)
2	NaoTH	3.79 (0.14)	1.85 (0.10)
4	BoldHearts	3.15 (0.13)	0.08 (0.11)
5-8	SEURedSun	1.93 (0.13)	1.16 (0.1)
5-8	RoboCanes	1.81 (0.12)	0.38 (0.09)
5-8	FCPortugal	1.57 (0.11)	**-0.43** (0.09)
9-16	UTAustinVilla	1.54 (0.09)	**-0.9** (0.09)
9-16	FutK	0.23 (0.06)	**-2.14** (0.1)
9-16	BeeStanbul	**-0.76** (0.07)	**-4.08** (0.11)
9-16	Nexus3D	**-1.67** (0.06)	**-4.08** (0.09)
9-16	Little Green Bats	**-1.84** (0.08)	**-5.0** (0.11)
9-16	NomoFC	**-3.62** (0.09)	**-7.07** (0.09)
17-20	Bahia3D	**-3.59** (0.08)	**-7.49** (0.1)
17-20	RoboPub	**-5.25** (0.08)	**-7.92** (0.1)
17-20	Alzahra	**-6.39** (0.08)	**-10.59** (0.09)

Goal Differential (stderr)

Future Work

- Extend the scope of learning within our agent:
  - Waiting times between frames
  - Replace hand-coded skills: fine positioning, getting up
  - Decision thresholds
- Alternative parameterizations: closed-loop, inverse kinematics
- Extend to real robots?

Related Work

- N. Hansen. *The CMA Evolution Strategy: A Tutorial, January 2009.*

- N. Shafii, L. P. Reis, and N. Lao. Biped walking using coronal and sagittal movements based on truncated Fourier series, January 2010.

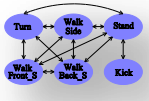
- J. E. Pratt. *Exploiting Inherent Robustness and Natural Dynamics in the Control of Bipedal Walking Robots. PhD thesis, Massachusetts* Institute of Technology, Cambridge, MA, USA, June 2000.

- N. Kohl and P. Stone. Machine learning for fast quadrupedal locomotion, 2004.

Summary

- We presented a learning architecture for a simulated humanoid robot soccer player

- Optimized over 100 parameters
- Used 2 ideas for improving speed while maintaining stability:
  - Optimizing under constraints
  - Skills decoupling



- A main building block in our agent, which is competitive with Robocup 2010 top-8 teams
- Found a new, successful application for the relatively new, CMA-ES algorithm