# UT Austin Villa 2011: RoboCup 3D Simulation League Champion

Patrick MacAlpine

Department of Computer Science, The University of Texas at Austin

April 15, 2012

#### What is RoboCup?

- International robotics competition founded in 1997
- Consists of many different robot soccer leagues
- Includes non-soccer robot competitions: RoboCup Rescue & RoboCup @Home



#### RoboCup Goal

Have a team of fully autonomous humanoid robot soccer players beat the human World Cup champions by 2050



Humans vs Robots

## RoboCup 3D Simulation Domain

- Teams of 9 vs 9 autonomous agents play soccer
- Realistic physics using Open Dynamics Engine (ODE)
- Agents modeled after Aldebaron Nao robot
- Agent receives noisy visual information about environment
- Agents can communicate with each other over limited bandwidth channel



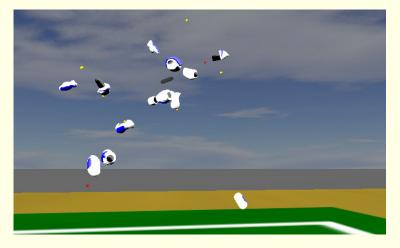


#### Advantages of 3D Simulation

- Good for testing both low-level control and high-level processes
- Allows for quick prototyping of robot models and behavior
- Can do large scale machine learning
- Simulated robots don't break...

## Advantages of 3D Simulation

- Good for testing both low-level control and high-level processes
- Allows for quick prototyping of robot models and behavior
- Can do large scale machine learning
- Simulated robots don't break... well at least not usually!



| RoboCup         | 2010          | 2011 |
|-----------------|---------------|------|
| Goals For:      | 11            |      |
| Goals Against:  | 17            |      |
| Record (W-L-T): | 4-5-1         |      |
| Place:          | Outside Top-8 |      |

| RoboCup         | 2010          | 2011 |
|-----------------|---------------|------|
| Goals For:      | 11            | 136  |
| Goals Against:  | 17            |      |
| Record (W-L-T): | 4-5-1         |      |
| Place:          | Outside Top-8 |      |

| RoboCup         | 2010          | 2011 |
|-----------------|---------------|------|
| Goals For:      | 11            | 136  |
| Goals Against:  | 17            | 0    |
| Record (W-L-T): | 4-5-1         |      |
| Place:          | Outside Top-8 |      |

| RoboCup         | 2010          | 2011   |
|-----------------|---------------|--------|
| Goals For:      | 11            | 136    |
| Goals Against:  | 17            | 0      |
| Record (W-L-T): | 4-5-1         | 24-0-0 |
| Place:          | Outside Top-8 |        |

| RoboCup         | 2010          | 2011   |
|-----------------|---------------|--------|
| Goals For:      | 11            | 136    |
| Goals Against:  | 17            | 0      |
| Record (W-L-T): | 4-5-1         | 24-0-0 |
| Place:          | Outside Top-8 | 1st    |

| RoboCup         | 2010          | 2011   |
|-----------------|---------------|--------|
| Goals For:      | 11            | 136    |
| Goals Against:  | 17            | 0      |
| Record (W-L-T): | 4-5-1         | 24-0-0 |
| Place:          | Outside Top-8 | 1st    |

# **BIG IMPROVEMENT!**

#### UT Austin Villa Team



Professor: Peter Stone











Graduate Students: Patrick MacAlpine, Daniel Urieli, Shivaram Kalyanakrishnan, Samuel Barrett, Yinon Bentor



Research Scientist: Michael Quinlan

Undergraduate Students: Francisco Barrera, Nick Collins, Adrian Lopez-Mobilia, Art Richards, Nicu Stiurca, Victor Vu

#### Talk Overview

- Overview of RoboCup
- 2. Omnidirectional Walk and Parameter Optimization
- 3. Kicking Engine
- 4. Dynamic Role Assignment and Positioning System
- 5. Analysis and Current/Future Work

#### Talk Overview

- 1. Overview of RoboCup
- 2. Omnidirectional Walk and Parameter Optimization
- 3. Kicking Engine
- 4. Dynamic Role Assignment and Positioning System
- 5. Analysis and Current/Future Work

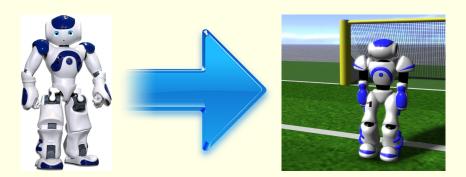
#### Motivation (2010 Walk)

- Consists of many fixed frame based skills
- Unable to quickly react (not omnidirectional)
- Not as stable as desired (completely open loop)



## Omnidirectional Walk Engine

- Double linear inverted pendulum model
- Based closely on that of walk engine by Graf et al
- Mostly open loop but not entirely
- Designed on actual Nao robot



#### Walk Engine Parameters

| Notation             | Description   |
|----------------------|---|
| maxStep <sub>i</sub> | Maximum step sizes allowed for $x$ , $y$ , and $\theta$ |
| <b>y</b> shift       | Side to side shift amount with no side velocity         |
| Z <sub>torso</sub>   | Height of the torso from the ground                     |
| Z <sub>step</sub>    | Maximum height of the foot from the ground              |
| 4                    | Fraction of a phase that the swing                      |
| f <sub>g</sub>       | foot spends on the ground before lifting                |
| fa                   | Fraction that the swing foot spends in the air          |
| fs                   | Fraction before the swing foot starts moving            |
| f <sub>m</sub>       | Fraction that the swing foot spends moving              |
| $\phi_{length}$      | Duration of a single step                               |
| δ                    | Factors of how fast the step sizes change               |
| <b>y</b> sep         | Separation between the feet                             |
| X <sub>offset</sub>  | Constant offset between the torso and feet              |
| ν.                   | Factor of the step size applied to                      |
| X <sub>factor</sub>  | the forwards position of the torso                      |
| err <sub>norm</sub>  | Maximum COM error before the steps are slowed           |
| err <sub>max</sub>   | Maximum COM error before all velocity reach 0           |

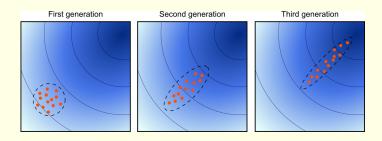
Parameters of the walk engine with the optimized parameters shown in bold

#### **Initial Walk Parameters**

- Designed and hand-tuned to work on the actual Nao robot
- Provides a slow and stable walk



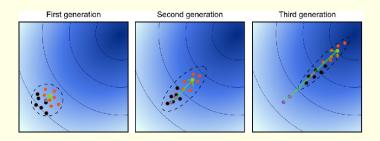
#### CMA-ES (Covariance Marix Adaptation Evolutionary Strategy)



- A stochastic, derivative-free, evolutionary numerical optimization method for non-linear or non-convex problems
- In 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

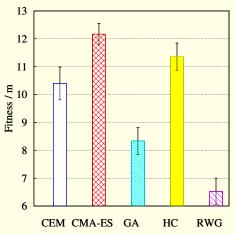
#### CMA-ES (Covariance Marix Adaptation Evolutionary Strategy)



- A stochastic, derivative-free, evolutionary numerical optimization method for non-linear or non-convex problems
- In 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

#### Learning Algorithms Evaluation



CEM Cross Entropy Method

CMA-ES Covariance Matrix Strategy Evolutionary Strategy

GA Gebetic Algorithm

**HC** Hill Climbing

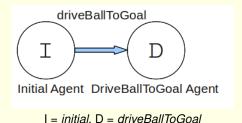
RWG Random Weight Guessing

#### Walk Control and Movement of Walk Engine

- Agent moves and turns in direction of target at the same time
  - ▶ Wins on average by .7 goals against non-turning agent
- When dribbling agent circles while always facing ball
- When told to stop agent jogs in place for half a second before entering fixed stand pose
  - Wins on average by .64 goals against non-jogging agent

### Drive Ball to Goal Optimization

- Agent given 30 seconds to dribble ball toward the goal
- Reward = distance ball travels toward goal averaged across 7 runs
- Parameters optimized using CMA-ES algorithm



#### Drive Ball to Goal Learned Parameters

- Wins by an average goal difference of 5.54 against Initial agent
- Wins by an average goal difference of 2.99 against 2010 walk agent



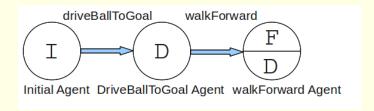
Agent with all learned 14 parameters performing driveBallToGoal optimization

### Problems with Drive Ball to Goal Optimization

- Agent not that fast
  - .43 m/s compared to .6 m/s speed of 2010 walk
- Agent unstable when stopping
- Agent overfits to when dribbling is going well

#### Walk Forward Optimization

- Agent walks forward for 10 seconds from a complete stop
- Reward = distance agent walks forward
- Faster speed of .78 m/s up from .43 m/s



I = initial. D = driveBallToGoal. F = walkForward

#### Walk Forward Agent Video



Attempts to transition between *driveBallToGoal* walk parameters (red 'D') and new *walkForward* parameters (yellow 'F')

### Go to Target Optimization

- Agent navigates to a series of target positions on the field
- Also have stop targets where agent is told to stop
- Reward: + for distance traveled toward target,
   for movement when told to stop

Fall = 5 if robot fell, 0 otherwise

 $d_{target}$  = distance traveled towards target

 $d_{moved}$  = total distance moved

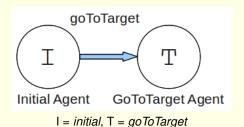
 $t_{total}$  = duration a target is active

 $t_{taken}$  = time taken to reach target, or  $t_{total}$  if target not reached

$$reward_{target} = d_{target} rac{t_{total}}{t_{taken}} - Fall$$
 $reward_{stop} = -d_{moved} - Fall$ 

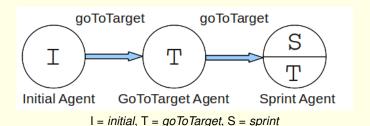
## Results of Go to Target Optimization

- Wins on average by 2.04 goals against DriveBallToGoal agent
- Walk speed increased to .64 m/s
- A little slow when positioning around the ball to dribble
- Good stability



### Sprint Parameter Set

- Learned with goToTarget optimization task
- Active when target is within 15° of agent's orientation
- gotoTarget parameter set is fixed during learning
- Walk speed increased to .71 m/s from .64 m/s
- Can switch on the fly between goto Target and sprint parameter sets



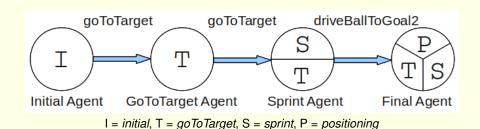
### Go to Target Optimization Video



Red 'T' = *gotoTarget* parameters, yellow 'S' = *sprint* parameters

# Drive Ball To Goal 2 Optimization

- Learn positioning parameter set active when .8 meters from the ball
- Dribble ball towards goal for 15 seconds from multiple different starting points around ball
- Reward = distance ball travels toward goal averaged across 16 runs
- goto Target and sprint parameter sets are fixed
- Wins by an average goal difference of .15 against Sprint agent



#### Drive Ball To Goal 2 Optimization Video



Red 'T' = *gotoTarget* parameters, yellow 'S' = *sprint* parameters, cyan 'P' = *positioning* parameters

#### Final Agent Video



Red 'T' = *gotoTarget* parameters, yellow 'S' = *sprint* parameters, cyan 'P' = *positioning* parameters

#### Walk Agent Performance

Game results of agents with different walk parameter sets. Entries show the average goal difference (row - column) from 100 ten minute games. Values in parentheses are the standard error.

|                 | Initial   | DriveBallToGoal | GoToTarget |
|-----------------|-----------|-----------------|------------|
| Final           | 8.84(.12) | 2.21(.12)       | .24(.08)   |
| GoToTarget      | 8.82(.11) | 2.04(.11)       |            |
| DriveBallToGoal | 5.54(.14) |                 |            |

### Walk in Competition



Action from the second half of the 2011 RoboCup 3D Simulation Final



# Omnidirectional Walk Optimization Summary

An optimization task should be representative of the overall task

# Omnidirectional Walk Optimization Summary

- An optimization task should be representative of the overall task
- Parameter sets can be combined but must be learned in conjunction with each other

# Omnidirectional Walk Optimization Summary

- An optimization task should be representative of the overall task
- Parameter sets can be combined but must be learned in conjunction with each other
- Machine learning is very effective for parameter optimization

#### Talk Overview

- 1. Overview of RoboCup
- 2. Omnidirectional Walk and Parameter Optimization
- 3. Kicking Engine
- 4. Dynamic Role Assignment and Positioning System
- 5. Analysis and Current/Future Work



# Kicking Engine Desired Properties

1. Agility: refers to taking shots quickly

### **Kicking Engine Desired Properties**

- 1. Agility: refers to taking shots quickly
- Robustness: entails taking accurate and powerful shots in spite of positioning errors (e.g., without the agent being perfectly lined up with the ball)

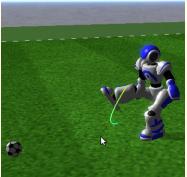
### Kicking Engine Desired Properties

- 1. Agility: refers to taking shots quickly
- Robustness: entails taking accurate and powerful shots in spite of positioning errors (e.g., without the agent being perfectly lined up with the ball)
- Versatility: refers to being able to kick in multiple directions from multiple ball starting locations

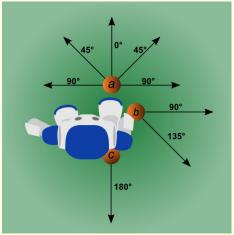
### Kick Engine Kinematics

- Define waypoints relative to ball for foot to reach
- Cubic Hermite splines used to compute path for foot to follow
- Inverse kinematics system determines if kick can be executed





#### **Directional Kicks**



The agent can dynamically kick the ball in varied directions with respect to the placement of the ball at a, b, and c



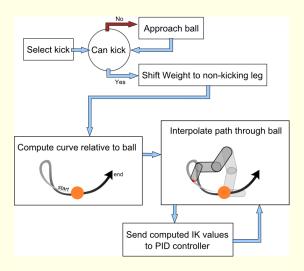
Different directional kicks

#### **Kick Choice**

 When approaching ball compute cost to execute each kick and choose kick with the lowest cost

```
\begin{array}{rcl} \textit{distCost} &=& |\text{agentPosition} - \text{targetOffsetPosition}| \ / m \\ \textit{turnCost} &=& \frac{|\text{agentOrientation} - \text{targetOrientation}|}{360^{\circ}} \\ \textit{ballPenalty} &=& \begin{cases} .5 & \text{if ball is in path to target offset} \\ 0 & \text{otherwise} \end{cases} \\ \textit{kickCost} &=& \text{distCost} + \text{turnCost} + \text{ballPenalty} \end{array}
```

### Kick Engine Workflow



### **Kick Optimization**

- Optimize parameters of kick: waypoint values, speed, ball offset
- Attempt ten kicks by approaching ball from different angles
- Reward = average distance ball travels toward target



#### Kick Performance

- Kicking agent loses by .15 goals on average to dribble only agent
- Strategy for best using kick not yet implemented (no passing yet)
- Shows improvement when used with an agent with a less effective walk (agent with initial walk parameters)
  - Kicking agent scored 8 goals while non-kicking agent failed to score when playing 100 games against each other

Inverse kinematics allows for robust and varied kicks

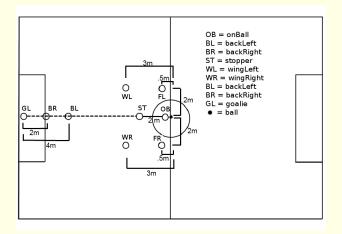
- Inverse kinematics allows for robust and varied kicks
- Machine learning is very effective for parameter tuning

- Inverse kinematics allows for robust and varied kicks
- Machine learning is very effective for parameter tuning
- Need appropriate strategy to effectively utilize kicking

#### Talk Overview

- 1. Overview of RoboCup
- 2. Omnidirectional Walk and Parameter Optimization
- 3. Kicking Engine
- 4. Dynamic Role Assignment and Positioning System
- 5. Analysis and Current/Future Work

#### **Formation**

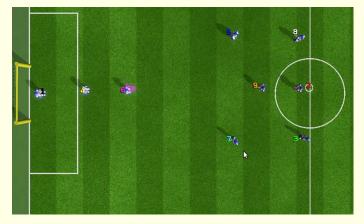


- Every player assigned to a role (position) on the field
- Positions based on offsets from ball or endline
- onBall role assigned to the player closest to the ball

Goalie positions itself independently

# Role Assignment Mapping and Assumptions

- One-to-one mapping of agents to positions
- Can be thought of as a role assignment function

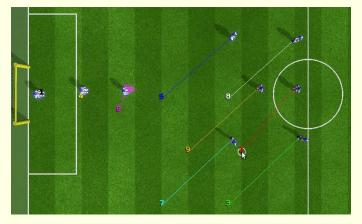


### Assumptions:

- 1. No two agents and no two roles occupy the same position
- 2. All agents move at constant speed along a straight line

# Role Assignment Mapping and Assumptions

- One-to-one mapping of agents to positions
- Can be thought of as a role assignment function



### Assumptions:

- 1. No two agents and no two roles occupy the same position
- 2. All agents move at constant speed along a straight line



# Desired Properties of a Role Assignment Function

 Minimizing longest distance - it minimizes the maximum distance from a player to target, with respect to all possible mappings

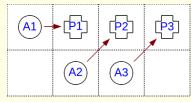
# Desired Properties of a Role Assignment Function

- 1. *Minimizing longest distance* it minimizes the maximum distance from a player to target, with respect to all possible mappings
- 2. Avoiding collisions agents do not collide with each other

# Desired Properties of a Role Assignment Function

- 1. *Minimizing longest distance* it minimizes the maximum distance from a player to target, with respect to all possible mappings
- 2. Avoiding collisions agents do not collide with each other
- 3. *Dynamically consistent* role assignments don't change or switch as agents move toward target positions

# Role Assignment Function $(f_v)$



Lowest lexicographical cost (shown with arrows) to highest cost ordering of mappings from agents (A1,A2,A3) to role positions (P1,P2,P3). Each row represents the cost of a single mapping.

- Mapping cost = vector of distances sorted in decreasing order
- Optimal mapping = lexicorgraphically sorted lowest cost mapping

# Validation of Role Assignment Function $f_{\nu}$

- f<sub>V</sub> minimizes the longest distance traveled by any agent (Property 1)
  as lexicographical ordering of distance tuples sorted in descending
  order ensures this.
- Triangle inequality will prevent two agents in a mapping from colliding (Property 2) it can be shown, as switching the two agents' targets reduces the maximum distance either must travel.
- f<sub>v</sub> is dynamically consistent (Property 3) as, under assumption all agents move toward their targets at the same constant rate, lowest cost lexicographical ordering of chosen mapping is preserved because distances between any agent and target will not decrease any faster than the distance between an agent and the target it is assigned to.

# Recursive Property of Role Assignment Function $f_V$

#### **Theorem**

Let A and P be sets of n agents and positions respectively. Denote the mapping  $m := f_v(A, P)$ . Let  $m_0$  be a subset of m that maps a subset of agents  $A_0 \subset A$  to a subset of positions  $P_0 \subset P$ . Then  $m_0$  is also the mapping returned by  $f_v(A_0, P_0)$ .

 Translation: Any subset of a lowest cost mapping is itself a lowest cost mapping

 If within any subset of a mapping a lower cost mapping is found, then the cost of the complete mapping can be reduced by augmenting the complete mapping with that of the subset's lower cost mapping

# Dynamic Programming Algorithm for Role Assignment

```
HashMap bestRoleMap = \varnothing

Agents = \{a_1, ..., a_n\}

Positions = \{p_1, ..., p_n\}

for k = 1 to n do

for all a in Agents do

S = \binom{n-1}{k-1} sets of k-1 agents from Agents - \{a\}

for all s in S do

Mapping m_0 = bestRoleMap[s]

Mapping m = (a \rightarrow p_k) \cup m_0

bestRoleMap[a \cup s] = mincost(m, bestRoleMap[a \cup s])

return bestRoleMap[Agents]
```

- Any subset of a lowest cost mapping is itself a lowest cost mapping
- Begin evaluating mappings of 1 agent and build up to n agents
- ullet Only evaluate mappings built from subset mappings returned by  $f_{\nu}$
- Evaluates  $n2^{n-1}$  mappings, for n = 8 is 1024 (brute force = 40,320)

### Positioning Video



Each position is shown as a color-coded number corresponding to the agent's uniform number assigned to that position. Agents update their role assignments and move to new positions as the ball or an agent is beamed (moved) to a new location.

# Positioning System Evaluation

| Team    | Goal Difference |
|---------|-----------------|
| Static  | .32 (.07)       |
| AllBall | .43 (.09)       |

Static Each role is statically assigned to an agent AllBall Every agent except goalie goes to the ball



# Positioning System Summary

Minimizing longest distance any agent travels is effective function

# Positioning System Summary

- Minimizing longest distance any agent travels is effective function
- Dynamic programming provides considerable increase in computational efficiency

# Positioning System Summary

- Minimizing longest distance any agent travels is effective function
- Dynamic programming provides considerable increase in computational efficiency
- No learning/optimization performed with positioning system...

#### Talk Overview

- 1. Overview of RoboCup
- 2. Omnidirectional Walk and Parameter Optimization
- 3. Kicking Engine
- 4. Dynamic Role Assignment and Positioning System
- 5. Analysis and Current/Future Work

### **Competition Analysis**

Average goal difference across 100 games against other agents in the competition

| Rank  | Team            | Goal Difference |
|-------|-----------------|-----------------|
| 3     | apollo3d        | 1.45 (.11)      |
| 5-8   | boldhearts      | 2.00 (0.11)     |
| 5-8   | robocanes       | 2.40 (0.10)     |
| 2     | cit3d           | 3.33 (0.12)     |
| 5-8   | fcportugal3d    | 3.75 (0.11)     |
| 9-12  | magmaoffenburg  | 4.77 (0.12)     |
| 9-12  | oxblue          | 4.83 (0.10)     |
| 4     | kylinsky        | 5.52 (0.14)     |
| 9-12  | dreamwing3d     | 6.22 (0.13)     |
| 5-8   | seuredsun       | 6.79 (0.13)     |
| 13-18 | karachikoalas   | 6.79 (0.09)     |
| 9-12  | beestanbul      | 7.12 (0.11)     |
| 13-18 | nexus3d         | 7.35 (0.13)     |
| 13-18 | hfutengine3d    | 7.37 (0.13)     |
| 13-18 | futk3d          | 7.90 (0.10)     |
| 13-18 | naoteamhumboldt | 8.13 (0.12)     |
| 19-22 | nomofc          | 10.14 (0.09)    |
| 13-18 | kaveh/rail      | 10.25 (0.10)    |
| 19-22 | bahia3d         | 11.01 (0.11)    |
| 19-22 | I3msim          | 11.16 (0.11)    |
| 19-22 | farzanegan      | 11.23 (0.12)    |

Across 2100 games played won all but 21 games which ended in ties (no losses)



## Additional Interesting Data

 Final competition agent beat agent with 2010 walk by an average goal difference of 6.32 goals across 100 games

## Additional Interesting Data

- Final competition agent beat agent with 2010 walk by an average goal difference of 6.32 goals across 100 games
- Agent with 2010 walk would have finished in tenth place

# Additional Interesting Data

- Final competition agent beat agent with 2010 walk by an average goal difference of 6.32 goals across 100 games
- Agent with 2010 walk would have finished in tenth place
- Number of times goalie touched the ball during the 2011 competition = 0



#### **Related Work**

- N. Hansen. The CMA Evolution Strategy: A Tutorial, January 2009.
- C. Graf, A. Härtl, T. Röefer, and T. Laue. A robust closed-loop gait for the standard platform league humanoid.
- 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.
- D. Urieli, P. MacAlpine, S. Kalyanakrishnan, Y. Bentor, and P. Stone. On optimizing interdependent skills: A case study in simulated 3d humanoid robot soccer, 2011.
- P. MacAlpine, D. Urieli, S. Barrett, S. Kalyanakrishnan, F. Barrera, A. Lopez-Mobilia, N. Stiurca, V. Vu, and P. Stone. UT Austin Villa 2011: A Winning Approach to the RoboCup 3D Soccer Simulation Competition, 2012.
- P. MacAlpine, S. Barrett, D. Urieli, V. Vu, and P. Stone. Design and Optimization of an Omnidirectional Humanoid Walk: A Winning Approach at the RoboCup 2011 3D Simulation Competition, 2012.
- P. Stone and M. Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, 1999.

#### **Related Work**

- N. Hansen. The CMA Evolution Strategy: A Tutorial, January 2009.
- C. Graf, A. Härtl, T. Röefer, and T. Laue. A robust closed-loop gait for the standard platform league humanoid.
- 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.
- D. Urieli, P. MacAlpine, S. Kalyanakrishnan, Y. Bentor, and P. Stone. On optimizing interdependent skills: A case study in simulated 3d humanoid robot soccer, 2011.
- P. MacAlpine, D. Urieli, S. Barrett, S. Kalyanakrishnan, F. Barrera, A. Lopez-Mobilia, N. Stiurca, V. Vu, and P. Stone. UT Austin Villa 2011: A Winning Approach to the RoboCup 3D Soccer Simulation Competition, 2012.
- P. MacAlpine, S. Barrett, D. Urieli, V. Vu, and P. Stone. Design and Optimization of an Omnidirectional Humanoid Walk: A Winning Approach at the RoboCup 2011 3D Simulation Competition, 2012.
- P. Stone and M. Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, 1999.

## Getup from Back

Series of fixed poses assumed sequentially



Getup from Back

### Getup from Front

Series of fixed poses assumed sequentially



Getup from Front

### Fully Holonomic Walk

Can walk in all directions with equal velocity



Fully Holonomic Walk



Kickoff 2011



Kickoff 2012



Kickoff 2012 Slow

# **Kickoff Optimization**





#### **Future Work**

- Attempt to apply learned walks in simulation to actual Nao robots
- Improve kicking strategy and add passing
- Attempt to learn better formations with machine learning
- Explore path planning ideas

 UT Austin Villa is a complete agent that won the 2011 RoboCup 3D simulation competition

- UT Austin Villa is a complete agent that won the 2011 RoboCup 3D simulation competition
- Key components of the agent are it's omnidirectional walk, kicking engine, and dynamic positioning system

- UT Austin Villa is a complete agent that won the 2011 RoboCup 3D simulation competition
- Key components of the agent are it's omnidirectional walk, kicking engine, and dynamic positioning system
- The omnidirectional walk proved to be the crucial component in winning the competition

- UT Austin Villa is a complete agent that won the 2011 RoboCup 3D simulation competition
- Key components of the agent are it's omnidirectional walk, kicking engine, and dynamic positioning system
- The omnidirectional walk proved to be the crucial component in winning the competition
- Optimizing parameters though machine learning is the underlying theme for the team's success

#### More Information

UT Austin Villa 3D Simulation Team homepage: www.cs.utexas.edu/~AustinVilla/sim/3dsimulation/

Email: patmac@cs.utexas.edu



This work has taken place in the Learning Agents Research Group (LARG) at UT Austin. LARG research is supported in part by NSF (IIS-0917122), ONR (N00014-09-1-0658), and the FHWA (DTFH61-07-H-00030).