Positioning to Win: A Dynamic Role Assignment and Formation Positioning System

Patrick MacAlpine, Francisco Barrera, and Peter Stone

Department of Computer Science, The University of Texas at Austin

June 24, 2012



2011 RoboCup 3D Simulation Domain

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





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

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

- 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

- 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

Positioning Video Brute Force



Positioning Video Brute Force



• Brute force requires evaluating n! mappings, for n = 8 is 40,320

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



{P1}	{P2,P1}	{P3,P2,P1}

- Begin evaluating mappings of 1 agent and build up to n agents
- Only evaluate mappings built from subset mappings returned by f_V
- Evaluates $n2^{n-1}$ mappings, for n = 8 is 1024 (brute force = 40,320)



{P1}	{P2,P1}	{P3,P2,P1}
A1→P1		
A2→P1		
A3→P1		

- Begin evaluating mappings of 1 agent and build up to n agents
- Only evaluate mappings built from subset mappings returned by f_{v}
- Evaluates $n2^{n-1}$ mappings, for n = 8 is 1024 (brute force = 40,320)



{P1}	{P2,P1}	{P3,P2,P1}
$A1 \rightarrow P1$	A1 \rightarrow P2, $f_{\nu}(A2\rightarrow$ P1)	
$A2 \rightarrow P1$	A1 \rightarrow P2, f_v (A3 \rightarrow P1)	
$A3 \rightarrow P1$	A2 \rightarrow P2, f_v (A1 \rightarrow P1)	
	A2 \rightarrow P2, f_{v} (A3 \rightarrow P1)	
	A3 \rightarrow P2, $f_{v}(A1\rightarrow$ P1)	
	A3 \rightarrow P2, $f_{\nu}(A2\rightarrow$ P1)	

- Begin evaluating mappings of 1 agent and build up to n agents
- Only evaluate mappings built from subset mappings returned by f_{v}
- Evaluates $n2^{n-1}$ mappings, for n = 8 is 1024 (brute force = 40,320)



{P1}	{P2,P1}	{P3,P2,P1}
A1→P1	A1 \rightarrow P2, $f_{\nu}(A2\rightarrow$ P1)	A1 \rightarrow P3, $f_{\nu}(\{A2,A3\}\rightarrow\{P1,P2\})$
A2→P1	A1 \rightarrow P2, f_v (A3 \rightarrow P1)	A2 \rightarrow P3, $f_{\nu}(\{A1,A3\}\rightarrow\{P1,P2\})$
A3→P1	A2 \rightarrow P2, f_v (A1 \rightarrow P1)	A3 \rightarrow P3, $f_{\nu}(\{A1,A2\}\rightarrow\{P1,P2\})$
	A2 \rightarrow P2, f_v (A3 \rightarrow P1)	
	A3 \rightarrow P2, $f_{v}(A1\rightarrow$ P1)	
	A3 \rightarrow P2, f_v (A2 \rightarrow P1)	

- Begin evaluating mappings of 1 agent and build up to n agents
- Only evaluate mappings built from subset mappings returned by f_v
- 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.

Voting Coordination System



- Each agent broadcasts ball position, own position, and suggested role mapping during allotted time slot
- Sliding window stored of mappings received over last *n* time slots evaluated and mapping with the most number of votes is chosen
- If two mappings both have greatest number of votes then tie breaker goes to mapping with most recent vote received

Voting Coordination System



- Each agent broadcasts ball position, own position, and suggested role mapping during allotted time slot
- Sliding window stored of mappings received over last *n* time slots evaluated and mapping with the most number of votes is chosen
- If two mappings both have greatest number of votes then tie breaker goes to mapping with most recent vote received
- Syncronization: With voting system = 100%, without = 36%

Positioning System Evaluation Agents

AllBall No formations and every agent except for the goalie goes to the ball.

NoCommunication Agents do not communicate with each other.

Static Role statically assigned to agents based on uniform number.

Offensive Offensive formation in which all agents except for the goalie are positioned in a close symmetric formation behind the ball.

NearestStopper The *stopper* role position is mapped to nearest agent.

PathCost Agents add in the cost of needing to walk around known obstacles (using collision avoidance), such as the ball and agent assuming the *onBall* role, when computing distances of agents to role positions.

Positioning System Evaluation

	UTAustinVilla	Apollo3D	CIT3D
Offensive	Offensive 0.21 (.09)		3.89 (.12)
AllBall	0.09 (.08)	1.69 (.13)	3.56 (.13)
PathCost	0.07 (.07)	1.27 (.11)	3.25 (.11)
NearestStopper	0.01 (.07)	1.26 (.11)	3.21 (.11)
UTAustinVilla	—	1.05 (.12)	3.10 (.12)
Static	-0.19 (.07)	0.81 (.13)	2.87 (.11)
NoCommunication	-0.30 (.06)	0.41 (.11)	1.94 (.10)

Positioning System Evaluation

	UTAustinVilla	Apollo3D	CIT3D
PositiveCombo	0.33 (.07)	2.16 (.11)	4.09 (.12)
Offensive	0.21 (.09)	1.80 (.12)	3.89 (.12)
AllBall	0.09 (.08)	1.69 (.13)	3.56 (.13)
PathCost	0.07 (.07)	1.27 (.11)	3.25 (.11)
NearestStopper	0.01 (.07)	1.26 (.11)	3.21 (.11)
UTAustinVilla	—	1.05 (.12)	3.10 (.12)
Static	-0.19 (.07)	0.81 (.13)	2.87 (.11)
NoCommunication	-0.30 (.06)	0.41 (.11)	1.94 (.10)

PositiveCombo Offensive + PathCost + NearestStopper agents

Positioning System Evaluation

	UTAustinVilla	Apollo3D	CIT3D
PositiveCombo	0.33 (.07)	2.16 (.11)	4.09 (.12)
Offensive	0.21 (.09)	1.80 (.12)	3.89 (.12)
AllBall	0.09 (.08)	1.69 (.13)	3.56 (.13)
PathCost	0.07 (.07)	1.27 (.11)	3.25 (.11)
NearestStopper	0.01 (.07)	1.26 (.11)	3.21 (.11)
UTAustinVilla	—	1.05 (.12)	3.10 (.12)
Static	-0.19 (.07)	0.81 (.13)	2.87 (.11)
NoCommunication	-0.30 (.06)	0.41 (.11)	1.94 (.10)

PositiveCombo Offensive + PathCost + NearestStopper agents

PositiveCombo agent beat AllBall agent by average of .31 goals. Record of 43 wins, 20 losses, 37 ties

• Minimizing longest distance any agent travels is effective function

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

- Minimizing longest distance any agent travels is effective function
- Dynamic programming provides considerable increase in computational efficiency
- Dynamic roles with an aggressive formation does the best

- Minimizing longest distance any agent travels is effective function
- Dynamic programming provides considerable increase in computational efficiency
- Dynamic roles with an aggressive formation does the best
- Communication and path planning are important

Related Work

- P. MacAlpine, D. Urieli, S. Barrett, S. Kalyanakrishnan, F. Barrera, A. Lopez-Mobilia, N. Ştiurcă, V. Vu, and P. Stone. UT Austin Villa 2011 RoboCup 3D Simulation Team Report, 2011.
- W. Chen and T. Chen. Multi-robot dynamic role assignment based on path cost, 2011.
- N. Lau, L. Lopes, G. Corrente, and N. Filipe. Multi-robot team coordination through roles, positionings and coordinated procedures, 2009.
- L. Reis, N. Lau, and E. Oliveira. Situation based strategic positioning for coordinating a team of homogeneous agents, 2001.
- P. Stone and M. Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork, 1999.

Future Work

- Implement passing and create formations to support this
- Attempt to learn better formations with machine learning
- Improve efficiency in calculating f_{V}
- Explore other role assignment functions
- Extensions for heterogenous agents

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).

```
HashMap bestRoleMap = \emptyset

Agents = {a<sub>1</sub>, ..., a<sub>n</sub>}

Positions = {p<sub>1</sub>, ..., p<sub>n</sub>}

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]
```

As $\binom{n-1}{k-1}$ agent subset mapping combinations are evaluated for mappings of each agent assigned to the *kth* position, the total number of mappings computed for each of the *n* agents is thus equivalent to the sum of the n-1 binomial coefficients. That is,

$$\sum_{k=1}^{n} \binom{n-1}{k-1} = \sum_{k=0}^{n-1} \binom{n-1}{k} = 2^{n-1}$$



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.

1:	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	$1 (A1 \rightarrow P1)$
2:	2 (A1 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A2→P1)
3:	$\sqrt{5}$ (A2 \rightarrow P3),	1 (A1→P1),	1 (A3→P2)
4:	$\sqrt{5}$ (A2 \rightarrow P3),	2 (A1→P2),	$\sqrt{2}$ (A3 \rightarrow P1)
5:	3 (A1→P3),	1 (A2→P1),	1 (A3→P2)
6:	3 (A1 \rightarrow P3).	$\sqrt{2}$ (A2 \rightarrow P2).	$\sqrt{2}$ (A3 \rightarrow P1)

(P1)	(P2,P1)	(P3,P2,P1)



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.

1:	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A1→P1)
2:	2 (A1→P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A2→P1)
3:	$\sqrt{5}$ (A2 \rightarrow P3),	1 (A1→P1),	1 (A3→P2)
4:	$\sqrt{5}$ (A2 \rightarrow P3),	2 (A1→P2),	$\sqrt{2}$ (A3 \rightarrow P1)
5:	3 (A1→P3),	1 (A2→P1),	1 (A3→P2)
6:	$3 (A1 \rightarrow P3),$	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P1)

(P1)	(P2,P1)	(P3,P2,P1)
A1→P1=(1)		
A2→P1=(1)		
A3→P1=(√2)		



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.

1:	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A1 \rightarrow P1)
2:	2 (A1 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A2→P1)
3:	$\sqrt{5}$ (A2 \rightarrow P3),	1 (A1 \rightarrow P1),	1 (A3→P2)
4:	$\sqrt{5}$ (A2 \rightarrow P3),	2 (A1→P2),	$\sqrt{2}$ (A3 \rightarrow P1)
5:	$3 (A1 \rightarrow P3),$	1 (A2→P1),	1 (A3→P2)
6:	$3 (A1 \rightarrow P3),$	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P1)

(P1)	(P2,P1)	(P3,P2,P1)
A1→P1=(1)	$A1 \rightarrow P2, A2 \rightarrow P1=(2,1)$	
$A2 \rightarrow P1=(1)$	A1 \rightarrow P2,A3 \rightarrow P1=(2, $\sqrt{2}$)	
A3 \rightarrow P1=($\sqrt{2}$)	$\begin{array}{c} A2 \rightarrow P2, A1 \rightarrow P1 = (\sqrt{2}, 1) \\ A2 \rightarrow P2, A3 \rightarrow P1 = (\sqrt{2}, \sqrt{2}) \end{array}$	
	$A3 \rightarrow P2, A1 \rightarrow P1=(1,1)$	
	$A3 \rightarrow P2, A2 \rightarrow P1 = (\sqrt{2}, 1)$	



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.

1:	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A1 \rightarrow P1)
2:	2 (A1 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A2→P1)
3:	$\sqrt{5}$ (A2 \rightarrow P3),	1 (A1 \rightarrow P1),	1 (A3→P2)
4:	$\sqrt{5}$ (A2 \rightarrow P3),	2 (A1→P2),	$\sqrt{2}$ (A3 \rightarrow P1)
5:	$3 (A1 \rightarrow P3),$	1 (A2→P1),	1 (A3→P2)
6:	$3 (A1 \rightarrow P3),$	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P1)

(P1)	(P2,P1)	(P3,P2,P1)
A1→P1=(1)	$A1 \rightarrow P2, A2 \rightarrow P1=(2,1)$	
A2→P1=(1)	A1 \rightarrow P2,A3 \rightarrow P1=(2, $\sqrt{2}$)	
A3 \rightarrow P1=($\sqrt{2}$)	$\begin{array}{c} A2 \rightarrow P2, A1 \rightarrow P1 = (\sqrt{2}, 1) \\ A2 \rightarrow P2, A3 \rightarrow P1 = (\sqrt{2}, \sqrt{2}) \end{array}$	
	$A3 \rightarrow P2, A1 \rightarrow P1=(1,1)$	
	$A3 \rightarrow P2, A2 \rightarrow P1 = (\sqrt{2}, 1)$	



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.

1:	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A1 \rightarrow P1)
2:	2 (A1 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A2→P1)
3:	$\sqrt{5}$ (A2 \rightarrow P3),	1 (A1 \rightarrow P1),	1 (A3→P2)
4:	$\sqrt{5}$ (A2 \rightarrow P3),	2 (A1 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P1)
5:	$3 (A1 \rightarrow P3),$	1 (A2→P1),	1 (A3→P2)
6:	$3 (A1 \rightarrow P3),$	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P1)

(P1)	(P2,P1)	(P3,P2,P1)
A1→P1=(1)		$A1 \rightarrow P3, A3 \rightarrow P2, A2 \rightarrow P1=(3, \sqrt{2}, 1)$
A2→P1=(1)		$A2 {\rightarrow} P3, A3 {\rightarrow} P2, A1 {\rightarrow} P1 {=} (\sqrt{5}, 1, 1)$
A3 \rightarrow P1=($\sqrt{2}$)	$A2 \rightarrow P2, A1 \rightarrow P1 = (\sqrt{2}, 1)$	A3 \rightarrow P3,A2 \rightarrow P2,A1 \rightarrow P1=($\sqrt{2},\sqrt{2},1$)
	$A3 \rightarrow P2, A1 \rightarrow P1=(1,1)$	
	$A3 \rightarrow P2, A2 \rightarrow P1 = (\sqrt{2}, 1)$	



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.

1:	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A1 \rightarrow P1)
2:	2 (A1 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P3),	1 (A2→P1)
3:	$\sqrt{5}$ (A2 \rightarrow P3),	1 (A1 \rightarrow P1),	1 (A3→P2)
4:	$\sqrt{5}$ (A2 \rightarrow P3),	2 (A1 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P1)
5:	$3 (A1 \rightarrow P3),$	1 (A2→P1),	1 (A3→P2)
6:	$3 (A1 \rightarrow P3),$	$\sqrt{2}$ (A2 \rightarrow P2),	$\sqrt{2}$ (A3 \rightarrow P1)

(P1)	(P2,P1)	(P3,P2,P1)
A1→P1=(1)		$A1 \rightarrow P3, A3 \rightarrow P2, A2 \rightarrow P1=(3, \sqrt{2}, 1)$
A2→P1=(1)		$A2 {\rightarrow} P3, A3 {\rightarrow} P2, A1 {\rightarrow} P1 {=} (\sqrt{5}, 1, 1)$
A3 \rightarrow P1=($\sqrt{2}$)	$A2 \rightarrow P2, A1 \rightarrow P1=(\sqrt{2}, 1)$	$A3 \rightarrow P3, A2 \rightarrow P2, A1 \rightarrow P1 = (\sqrt{2}, \sqrt{2}, 1)$
	$A3 \rightarrow P2, A1 \rightarrow P1=(1,1)$	
	$A3 \rightarrow P2, A2 \rightarrow P1 = (\sqrt{2}, 1)$	

Set Plays



Validation of Role Assignment Function f_v

- *f_v* 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_v* 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.

Validation of Role Assignment Function fv



Example collision scenario. If the mapping (A1-P2,A2-P1) is chosen the agents will follow the dotted paths and collide at the point marked with a C. Instead f_v will choose the mapping (A1-P1,A2-P2) as this minimizes maximum path distances and the agents will follow the path denoted by the solid arrows thereby avoiding the collision.

• If two agents in a mapping are to collide (Property 2) it can be shown, through the triangle inequality, that f_v will find a lower cost mapping as switching the two agents' targets reduces the maximum distance either must travel.

Validation of Role Assignment Function f_v Continued

- It is trivial to see that f_v minimizes the longest distance traveled by any agent (Property 1) as the lexicographical ordering of distance tuples sorted in descending order ensures this.
- As we assume all agents move toward their targets at the same constant rate, the distance between any agent and target will not decrease any faster than the distance between an agent and the target it is assigned to. This serves to preserve the lowest cost lexicographical ordering of the chosen mapping by f_v across all timesteps thereby providing dynamic consistency (Property 3)

Other Role Assignment Functions



Figure: Example where minimizing the sum of path distances fails to hold desired properties. Both mappings of $(A1 \rightarrow P1, A2 \rightarrow P2)$ and $(A1 \rightarrow P2, A2 \rightarrow P1)$ have a sum of distances value of 8. The mapping $(A1 \rightarrow P2, A2 \rightarrow P1)$ will result in a collision and has a longer maximum distance of 6 than the mapping $(A1 \rightarrow P1, A2 \rightarrow P2)$ whose maximum distance is 4. Once a mapping is chosen and the agents start moving the sum of distances of the two mappings will remain equal which could result in thrashing between the two.



Figure: Example where minimizing the sum of path distances squared fails to hold desired property of minimizing the time for all agents to have reached their target destinations. The mapping $(A1 \rightarrow P1, A2 \rightarrow P2)$ has a path distance squared sum of 19 which is less than the mapping $(A1 \rightarrow P2, A2 \rightarrow P1)$ for which this sum is 27. f_v will choose the mapping with the greater sum as its maximum path distance (proportional to the time for all agents to have reached their targets) is $\sqrt{17}$ which is less than the other mapping's maximum path distance of $\sqrt{18}$.

Collision Avoidance



Proximity Thresh Move at angle tangent to obstacle

Collision Thresh Move along vector combination of angle tangent to and 180° from obstacle

Collision Avoidance Video



Clear Path Unblocked path to target Blocked Path Blocked path to target Corrected Path Path to avoid obstacle Proximity Thresh Proximity threshold around obstacle Collision Thresh Collision threshold around obstacle