

# Gliders2012: Development and Competition Results

Edward Moore<sup>1\*</sup>, Oliver Obst<sup>1</sup>, Mikhail Prokopenko<sup>1</sup>, and Peter Wang<sup>1</sup>  
Jason Held<sup>2</sup>

<sup>1</sup> CSIRO Information and Communication Technologies Centre, Adaptive Systems,  
PO Box 76, Epping, NSW 1710, Australia

<sup>2</sup> Saber Astronautics Australia, 53 Balfour St, Chippendale, NSW 2008, Australia

**Abstract.** The RoboCup 2D Simulation League incorporates several challenging features, setting a benchmark for Artificial Intelligence (AI). In this paper we describe some of the ideas and tools around the development of our team, Gliders2012. In our description, we focus on the evaluation function as one of our central mechanisms for action selection. We also point to a new framework for watching log files in a web browser that we release for use and further development by the RoboCup community. Finally, we also summarize results of the group and final matches we played during RoboCup 2012, with Gliders2012 finishing 4th out of 19 teams.

## 1 Introduction

The RoboCup Simulation League [10] incorporates several challenging features, setting a benchmark for Artificial Intelligence (AI). The following list includes some of the most prominent characteristics of the RoboCup 2D Simulation League:

- distributed client/server system running on a network, leading to fragmented, localized and imprecise (noisy and latent) information about the environment (field) [12];
- concurrent communication with a medium-sized number of agents [21];
- heterogeneous sensory data (visual, auditory, kinetic) and limited range of basic commands/actuators (turn, kick, dash, ...) [19];
- asynchronous perception-action activity and limited window of opportunity to perform an action [6];
- autonomous decision-making under constraints enforced by teamwork (collaboration) and opponent (competition) [20];
- conflicts between reactivity and deliberation [18];
- no centralized controllers and centralized world model (no global vision, etc.) [16,17].

From the onset of the RoboCup effort it was recognized that, as a benchmark, RoboCup is fairly different from another classical AI problem — chess. As pointed out by Asada et al. [4], chess and RoboCup differ in a few key elements: environment (static vs dynamic), state change (turn-taking vs real-time), information accessibility (complete vs incomplete), sensor readings (symbolic vs non-symbolic), and control (central vs distributed). This difference has been well understood over the last decade. Nevertheless, there are some similarities, for example, efficient evaluation functions used by the RoboCup agents are conceptually similar

---

\* CSIRO authors are listed in alphabetical order.

to evaluation functions used by chess computers: in either case the agent is attempting to consider multiple future states, assign some values to the alternative outcomes, and choose an action optimizing the evaluations. One may argue that superior performance of recent world champions in the RoboCup 2D Simulation League [3,5] may be attributed, at least partially, to sophisticated evaluation functions employed by these teams. In this short paper we describe a novel mechanism utilizing action-dependent evaluation functions, comparing it to some well known constructive models used by belief revision and belief update [14].

The experiments are carried out using a new simulated soccer team for the RoboCup soccer 2D simulator [7], Gliders2012. The team code is written by C++ using agent2d: the well-known base code developed by Akiyama et al. [1]. Other software packages are used as well:

- librcsc: a base library for RCSS with various utilities describing relevant geometrical constructs, world model, etc.;
- soccerwindow2: a viewer program for RCSS, working as a monitor client, a log player and a visual debugger;
- fedit2: a formation editor for agent2d, allowing to design a team formation.

## 2 Motivation and approach

### 2.1 Chess analogy

As argued by Laramée, in chess “the evaluation function, is unique in a very real sense: while search techniques are pretty much universal and move generation can be deduced from a game’s rules and no more, evaluation requires a deep and thorough analysis of strategy” [11]. He lists several main board evaluation metrics: material balance (an account of which pieces are on the board for each side), mobility (a measure of how many move options are available, especially for powerful chess pieces), board control (a side controls a square if it has more pieces attacking it than the opponent), development (minor pieces should be brought into the game as quickly as possible), pawn formations, king safety and tropism (a measure of how easy it is for a piece to attack the opposing king; usually measured in terms of distance).

One may draw some parallels with RoboCup Simulation. For example, the goal safety and distances to the opposing goal are analogous to king safety and tropism, pawn formations may give some hints to team formations, development is somewhat similar to developing an attack from within your own half, board control is akin to blocking and marking opponent players (i.e., field control), mobility is achieved by either positioning teammates to receive a pass, or creating multiple directions for dribble, and material balance can be computed by accounting for heterogeneous player types and remaining stamina values. All these analogies are, of course, not direct — nevertheless, they may provide some inspiration for an evaluation function relevant for RoboCup Simulation.

### 2.2 Basic evaluation

The evaluation function of agent2D is, however, quite simple. Using the chess analogy, it implements *tropism* only, and is intended to make the basic client play in a goal-oriented

fashion. For a player controlling the ball, it considers two features of each possible resultant state  $s$ : its  $X$ -coordinate (the larger the better) and the distance from it to the opponent's goal (the smaller the better). That is, the opponent's goal is the ultimate desirable resultant state  $S$ , and each action  $a$  is rated in terms of a single distance metric  $D$

$$r(a) = D(s = \text{result}(a), S) . \quad (1)$$

The action that is selected is simply the one that minimizes the distance between resultant and desirable states, i.e., minimizes this metric:

$$a^* = \arg \min_a r(a) . \quad (2)$$

For the players who are not controlling the ball and are engaged in intercept behavior, the evaluation function is not specified explicitly. These players select positions on the field according to their roles in the team formation.

The evaluation function (2) has reached a significant aim: all types of actions (dribbles, passes, etc.) can be directly compared to each other in terms of a single metric. At the same time, the simple computation is not adequate to support a very sophisticated tactical play in mid-field, or even near opponent's penalty area. Another drawback is that all the actions are judged in relation to a single point: the opponent goal.

### 2.3 Multiple desirable states: tactics

Our main objective was to retain the advantage of a single metric, but diversify the evaluation by considering multiple points as desirable states. Moreover, we suggested [15] not only that the most desirable state can change from one cycle to another, but also that a player may entertain multiple desirable states at any given time (cycle). This diversity is brought about by different tactics. For example, a player may consider one desirable state  $S_1$  if passing to the left (action  $a_1$ ) pursuing one tactic, another resultant state  $S_2$  if passing to the right (action  $a_2$ ) guided by another tactic, and yet another desirable state  $S_3$  if dribbling to the center (action  $a_3$ ) suggested by a third tactic. Each of the considered actions is evaluated with respect to the corresponding desirable state that represents one of possible tactical ways to develop the play.

In other words, at any given time, there is a number  $m$  of tactics represented by a set of desirable states:  $\{S_1, \dots, S_m\}$ , and the feasible actions are partitioned into  $m$  sets:  $A_1, \dots, A_m$ , so that for every action  $a_i$ , there is a set  $A_j$  such that  $a_i \in A_j$ . We denote the function mapping an action to its tactical state by

$$\text{tactics} : a \rightarrow S . \quad (3)$$

Then each feasible action is rated with respect to the corresponding desirable state:

$$r(a) = D(s = \text{result}(a), S = \text{tactics}(a)) \quad (4)$$

followed by selection according to the optimization (2). This approach does not impose tactics in a top-down fashion, selecting one tactic and the sub-selecting the best action for the chosen tactic. Rather, all feasible actions are considered, and tactics contribute to the

evaluation via the desirable states suggested by the tactics. In certain cases the opponent’s goal becomes one of possible desirable states (one of the tactics), keeping the goal-oriented behavior of agents.

The difference between definitions (1) and (4) is simply that the desirable states that the player is trying to reach are not independent of actions, but rather *are* action-dependent, and this dependence is tactical. To re-iterate, the comparison between two actions  $a_1$  and  $a_2$  according to the first definition (1) always assumes the same action-independent state  $S$  that is evaluated against, while the proposed definition (4) allows for different desirable states  $S_1 = \text{tactics}(a_1)$  and  $S_2 = \text{tactics}(a_2)$ . The metric  $D$  is the same for all actions, retaining the advantage of a uniform comparison across different action types.

We would like to point out at this stage a difference between the proposed action-dependent evaluation function and other action-dependent formalisms, e.g., with action-dependent features generalizing state space proposed by Stone and Veloso [22]. The latter study described a multi-agent learning paradigm called team-partitioned, opaque-transition reinforcement learning (TPOT-RL). TPOT-RL introduced the concept of using action-dependent features to generalize the state space. However, regardless of action-dependent features, each possible action  $a$  is evaluated by TPOT-RL based on the current state of the world using a *fixed* function  $e : (S, A) \rightarrow U$ . That is, the function  $e$  is the same for all actions in TPOT-RL.

Another interesting point is the analogy between multiple desirable states unified by the proposed evaluation function and the constructive model for belief update and belief revision [14]. Belief revision is the process by which a rational agent changes their beliefs about a static world in the light of new data. Belief update on the other hand is the process by which an agent maintains their beliefs up to date with an evolving world. The constructive model for belief revision includes a single similarity structure centered on all possible worlds consistent with current beliefs (a single system of nested spheres), and identifies the nearest sphere which is consistent with the new data. Peppas et al. [14] have shown that the model for belief update uses multiple systems of spheres (one for each possible world), finds in parallel the spheres consistent with the new data that are nearest to their respective central possible worlds, and collects possible worlds within these spheres. Arguably, the action-dependent evaluation proposed here is akin to the constructive model of belief update.

## 2.4 Mobility and field control

The function *tactics* implements the *mobility* aspect of evaluation, by diversifying options of the player controlling the ball in continuing the game. The other teammates can also use this function in selecting a desirable state for their positioning. That is, a player choosing a position on the field does not have to have a single best point, given the current state. It may consider multiple points, each of which is again dependent on the action. For example, the player may consider state (point)  $S_1$  when moving to the left wing with action  $a_1$ , and state (point)  $S_2$  when blocking a nearest opponent with action  $a_2$ . Each of the resultant states  $s_1 = \text{result}(a_1)$  and  $s_2 = \text{result}(a_2)$  are compared with the corresponding desirable states suggested by the tactics  $S_1$  and  $S_2$ , and the action achieving the best proximity in terms of the metric  $D$  is selected. The diversification in positioning achieves both *mobility* (by enabling better passes to these teammates) and *field control* — by taking key points and blocking key directions.

The idea of field control can be traced to a generic framework describing abstract spatio-temporal relationships described by Dylla et al. [8]. The latter work did not mention field control explicitly but argued that a reachability relation is needed to express spatial relationships between the players and the ball. They suggested to use Voronoi diagrams: a Voronoi diagram is the partitioning of a plane with  $n$  points into  $n$  convex polygons such that each polygon contains exactly one point and every point in the given polygon is closer to its central point than any other [8]. This was further developed by Akiyama et al. who used a dual representation of Voronoi diagrams — the Delaunay triangulation [2,3].

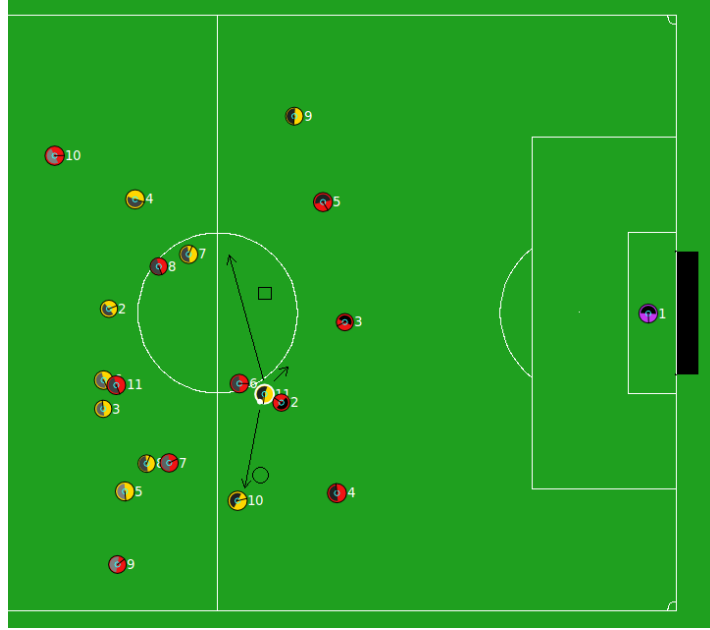
## 2.5 Example

Figure 1 illustrates the concept of action-dependent evaluation. The player controlling the ball (left team, number 11) has several options available: it can dribble in a general forward-left direction, pass to teammates 7 and 10 (in a number of ways, including direct and lead passes), etc. We consider three choices (shown by arrows): dribble forward-left, pass to the left to teammate 7, and pass to the right to teammate 10. The agent2d’s evaluation function would most likely rate the dribble higher, as the resultant state (the arrow-head) has a larger  $X$ -coordinate and a smaller distance from the opponent’s goal than the alternatives. The new evaluation function identifies two desirable states instead, shown by a small rectangle to the left of player 11, and a small circle to its right. The rectangle defines the tactic suggesting to develop an attack to the left and through the center, and the circle corresponds to the tactic preferring the right wing. The dribble and pass to teammate 7 are partitioned to the first tactic (rectangle), and the pass to teammate 10 belongs to the second tactic (circle). Each of these actions is rated by proximity of their resultant states (the arrow-heads) to the rectangle and circle respectively. The pass to number 10 has the smaller distance between the resultant and desirable states, and is then selected.

## 3 Development and Results

The proposed approach was implemented in Gliders2012 — a new team based on agent2d [1]. We carried out multiple iterative experiments, matching Gliders2012 up against the agent2d (HELIOS Base team), and achieving  $\approx +4.0$  goal difference, typically averaged over 100 games. In doing that, some of the most important tools for development were a set of scripts to automate running tournaments. Average results of such a tournament are a useful indicator if a proposed change in the team is really helpful or possibly hurts performance. Oftentimes, it is helpful to briefly watch some of the games that are stored on the simulation server. To make these matches available for viewing over web browser, the current existing solution is to convert them into flash (SWF) format. Even though source code for this program is available [9], a required library to create SWF files is proprietary and not anymore distributed. As a result, this option is not available for new teams.

We created a new set of programs to visualize matches in a browser window. This time, the main implementation uses HTML5 / javascript, with a pre-processing step to create simple-to-parse log files that can be conveniently transferred over the internet. The current state of the visualization is basic but functional, and released by us to the RoboCup community



**Fig. 1.** Action-dependent evaluation. Small rectangle and circle show different desirable states. The arrows point to possible resultant states. The pass to player 10 is selected since the distance between its resultant and desirable states is the smallest.

in the hope that it will be improved over time, re-released, and be used for showing games to general public in future competitions. Figure 2 shows a screenshot.

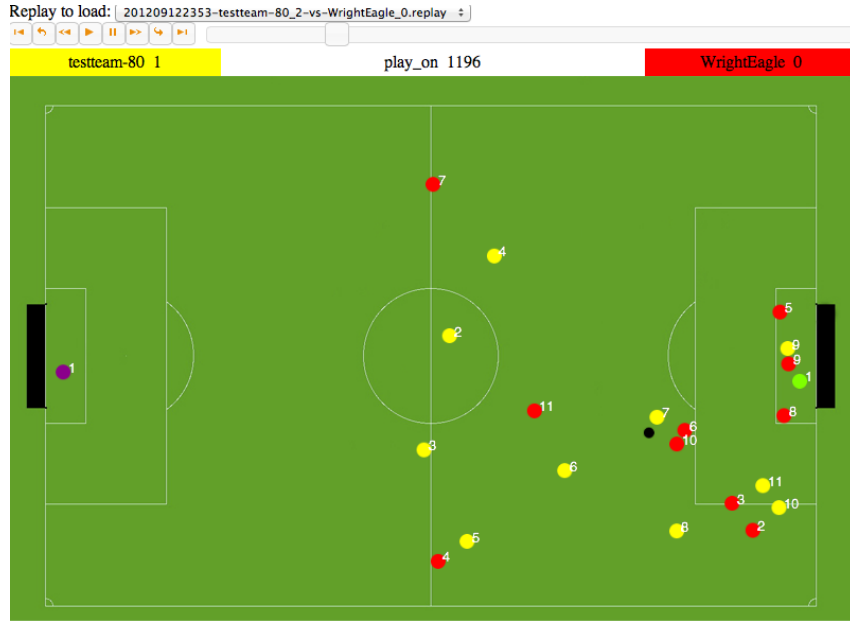
The steps required to produce a HTML5 visualization from a stored log file (\*.rcg.gz) are as follows:

1. Convert the log file into an intermediate text file. The required tool for this is distributed with the soccer server (`rcg2txt`).
2. Convert the intermediate text file into a \*.replay file. This is also a text file, but with only the essential information to display the match.
3. The \*.replay file then can be loaded from a PHP / HTML file over a browser. The entire log player consists of a PHP / HTML script and a few javascript and CSS files.

The log player is available (along with the necessary converter) at the Gliders2012 web page [www.oliverobst.eu/research/robotics-gliders2012-simulation-league-robocup-team](http://www.oliverobst.eu/research/robotics-gliders2012-simulation-league-robocup-team).

### 3.1 Results at RoboCup 2012

Eventually, our team also participated in RoboCup 2012 with convincing results. The tournament was played over several rounds, where Gliders2012 proceeded to the semi-finals, resulting in a 4th place. Our detailed results are as follows:



**Fig. 2.** HTML5 Logplayer developed by us. To reduce file transmission times, log files are converted into a compact textual representation, and can be selected using a HTML drop down menu.

**Seeding round (Round 0)** A seeding round was played, with the 18 participating teams distributed over 4 groups of 4 or 5 teams each. The results of these groups were to be used to seed all teams more fairly for the subsequent round 1.

Gliders2012 were allocated to group C with 4 other teams: GPR-2D (Brazil), MarLik (Iran), AUT\_2D (Iran), and Riton (also Iran). Gliders2012 ended up leading the group, with one draw against MarLik, and all other matches won (Table 1).

**Round 1** All teams were to proceed to round 1. Round 1 was played in two groups of 9 (groups E and F), with Gliders2012 in group E. Other teams in group E were: Warthog, GPR-2D, and ITAndroids (all Brazil), Oxxy (Romania), GDUT\_TiJi, YuShan2012, and WrightEagle (all China), as well as Riton (Iran).

Gliders2012 won 3 matches, drew 3, and lost 2, resulting in a 3rd place ranking in this group (Table 2).

**Round 2** The top 6 teams of both round 1 groups proceeded to round 2, groups G and H. Gliders2012 played in group G, against Riton and MarLik (Iran), FCPortugal (Portugal), robOTTO (Germany), and WrightEagle (China). Gliders2012 won 3 matches, and lost 2, and ended up ranked 4th with an equal number of points to the second and third place.

**Round 3** The top 4 teams of both groups in round 2 proceeded to round 3, groups I and J. The group I teams were HELIOS2012 (Japan), Gliders2012, AUT\_2D (Iran), and robOTTO

**Table 1.** Round 0, group C results.

Place	Team	Points	Total Score	W	D	L
1	Gliders2012	10	9:2	3	1	0
2	MarliK	8	4:2	2	2	0
3	GPR-2D	4	2:3	1	1	2
4	AUT_2D	4	2:3	1	1	2
5	Riton	1	0:7	0	1	3

**Table 2.** Round 1, group E results.

Place	Team	Points	Total Score	W	D	L
1	WrightEagle	24	53 : 7	8	0	0
2	YuShan2012	15	15 : 12	4	3	1
3	Gliders2012	12	18 : 17	3	3	2
4	ITAndroids	11	13 : 19	3	2	3
5	Riton	10	10 : 10	2	4	2
6	GDUT_TiJi	10	18 : 19	3	1	4
7	GPR-2D	9	5 : 10	2	3	3
8	Oxxy	6	10 : 27	2	0	6
9	Warthog	2	4 : 25	0	2	6

**Table 3.** Round 2, group G results.

Place	Team	Points	Total Score	W	D	L
1	WrightEagle	15	19 : 3	5	0	0
2	robOTTO	9	9 : 7	3	0	2
3	MarliK	9	7 : 8	3	0	2
4	Gliders2012	9	9 : 11	3	0	2
5	FCPortugal	3	7 : 11	1	0	4
6	Riton	0	8 : 19	0	0	5

**Table 4.** Round 3, group I results.

Place	Team	Points	Total Score	W	D	L
1	HELIOS2012	9	4 : 0	3	0	0
2	Gliders2012	6	5 : 1	2	0	1
3	AUT_2D	3	1 : 3	1	0	2
4	robOTTO	0	0 : 6	0	0	3

(Germany). With two won matches and one lost match in this group, Gliders2012 proceeded to the semi-finals.

**Semi-Finals and 3rd place match** Gliders played WrightEagle for the semi-finals (2 matches), both 0:2. The subsequent match for the third place was lost against MarliK (0:1). HELIOS2012 won the tournament, WrightEagle became runner-up.

## 4 Conclusion

We described a novel mechanism utilizing action-dependent evaluation functions, having applied it in the RoboCup Simulation 2D. The mechanism can be contrasted with some well known constructive models used by belief revision and belief update [14]. The approach also allowed us to draw parallels with evaluation functions employed by chess-playing computers, in terms of mobility, field control, tropism, etc. The evaluation function that varies desirable states dependent on contemplated actions is applicable in both ball-controlling and positioning scenarios. The tactics that correspond to multiple desirable states are not imposed in a top-down fashion, but rather contribute to the evaluation via these desirable states.

Our proposed approach and its implementation has shown to be successful both in many experiments as well as during tournament.



**Acknowledgments** The Authors are thankful to Valentina Cupac, Andrew Curline, Tim D’Adam, Ivan Duong, James Nugent, Tom Stewart for their contribution. Team logo was created by Matthew Chadwick. Some of the Authors have been involved with RoboCup Simulation 2D in the past, however the code of their previous teams (Cyberroos and RoboLog, see, e.g., [17,13]) is not used in Gliders2012.

## References

1. Hidehisa Akiyama. Agent2D Base Code. <http://www.rctools.sourceforge.jp>, 2010.
2. Hidehisa Akiyama and Itsuki Noda. Multi-agent positioning mechanism in the dynamic environment. In Ubbo Visser, Fernando Ribeiro, Takeshi Ohashi, and Frank Dellaert, editors, *RoboCup 2007: Robot Soccer World Cup XI*, pages 377–384. Springer, Berlin, Heidelberg, 2008.
3. Hidehisa Akiyama and Hiroki Shimora. Helios2010 team description. In *RoboCup 2010: Robot Soccer World Cup XIV*, volume 6556 of *Lecture Notes in Computer Science*. Springer, 2011.
4. Minoru Asada, Hiroaki Kitano, Itsuki Noda, and Manuela Veloso. RoboCup: Today and tomorrow – What we have have learned. *Artificial Intelligence*, 110:193–214, 1999.
5. Aijun Bai, Xiaoping Chen, Patrick MacAlpine, Daniel Urieli, Samuel Barrett, and Peter Stone. Wrighteagle and ut austin villa: Robocup 2011 simulation league champions. In *RoboCup 2011: Robot Soccer World Cup XV*, Lecture Notes in Artificial Intelligence. Springer, 2012.
6. Marc Butler, Mikhail Prokopenko, and Thomas Howard. Flexible synchronisation within RoboCup environment: A comparative analysis. In *RoboCup 2000: Robot Soccer World Cup IV*, pages 119–128, London, UK, 2001. Springer.
7. Mao Chen, Klaus Dorer, Ehsan Foroughi, Fredrick Heintz, ZhanXiang Huang, Spiros Kapetanakis, Kostas Kostiadis, Johan Kummeneje, Jan Murray, Itsuki Noda, Oliver Obst, Pat Riley, Timo Steffens, Yi Wang, and Xiang Yin. *Users Manual: RoboCup Soccer Server — for Soccer Server Version 7.07 and Later*. The RoboCup Federation, February 2003.
8. Frank Dylla, Alexander Ferrein, Gerhard Lakemeyer, Jan Murray, Oliver Obst, Thomas Röfer, Stefan Schiffer, Frieder Stolzenburg, Ubbo Visser, and Thomas Wagner. *Computers in Sport*, chapter Approaching a Formal Soccer Theory from the Behavior Specification in Robotic Soccer, pages 161–186. Bioengineering. WIT Press, 2008.
9. Thilo Girmann and Oliver Obst. robocup2flash version 0.3. <http://robolog.cvs.sourceforge.net/viewvc/robolog/robocup2flash/>, 2003.
10. Hiroaki Kitano, Milind Tambe, Peter Stone, Manuela M. Veloso, Silvia Coradeschi, Eiichi Osawa, Hitoshi Matsubara, Itsuki Noda, and Minoru Asada. The RoboCup Synthetic Agent Challenge 97. In *RoboCup-97: Robot Soccer World Cup I*, pages 62–73, London, UK, 1998. Springer.
11. François Dominic Laramée. Chess Programming Part VI: Evaluation Functions. [http://www.gamedev.net/page/resources/\\_/technical/artificial-intelligence/chess-programming-part-vi-evaluation-functions-r1208](http://www.gamedev.net/page/resources/_/technical/artificial-intelligence/chess-programming-part-vi-evaluation-functions-r1208), 2000.
12. Itsuki Noda and Peter Stone. The RoboCup Soccer Server and CMUnited Clients: Implemented Infrastructure for MAS Research. *Autonomous Agents and Multi-Agent Systems*, 7(1–2):101–120, July–September 2003.
13. Oliver Obst and Joschka Boedecker. Flexible coordination of multiagent team behavior using HTN planning. In Itsuki Noda, Adam Jacoff, Ansgar Bredendfeld, and Yasutake Takahashi, editors, *RoboCup 2005: Robot Soccer World Cup IX*, Lecture Notes in Artificial Intelligence, pages 521–528. Springer, Berlin, Heidelberg, New York, 2006.
14. Pavlos Peppas, Abhaya C. Nayak, Maurice Pagnucco, Norman Y. Foo, Rex Bing Hung Kwok, and Mikhail Prokopenko. Revision vs. update: Taking a closer look. In Wolfgang Wahlster, editor, *12th European Conference on Artificial Intelligence, Budapest, Hungary, August 11-16, 1996, Proceedings*, pages 95–99. John Wiley and Sons, Chichester, 1996.

15. Mikhail Prokopenko, Oliver Obst, Peter Wang, and Jason Held. Gliders2012: Tactics with action-dependent evaluation functions. In *RoboCup 2012: Team Description Paper*, Mexico City, Mexico, June 2012.
16. Mikhail Prokopenko and Peter Wang. Relating the entropy of joint beliefs to multi-agent coordination. In Gal A. Kaminka, Pedro U. Lima, and Raúl Rojas, editors, *RoboCup 2002: Robot Soccer World Cup VI*, volume 2752 of *Lecture Notes in Computer Science*, pages 367–374. Springer, 2003.
17. Mikhail Prokopenko and Peter Wang. Evaluating team performance at the edge of chaos. In Daniel Polani, Brett Browning, Andrea Bonarini, and Kazuo Yoshida, editors, *RoboCup 2003: Robot Soccer World Cup VII*, volume 3020 of *Lecture Notes in Computer Science*, pages 89–101. Springer, 2004.
18. Luís Paulo Reis, Nuno Lau, and Eugenio Oliveira. Situation based strategic positioning for coordinating a team of homogeneous agents. In *Balancing Reactivity and Social Deliberation in Multi-Agent Systems, From RoboCup to Real-World Applications (selected papers from the ECAI 2000 Workshop and additional contributions)*, pages 175–197, London, UK, 2001. Springer.
19. Patrick Riley, Peter Stone, and Manuela Veloso. Layered disclosure: Revealing agents’ internals. In C. Castelfranchi and Y. Lesperance, editors, *Intelligent Agents VII. Agent Theories, Architectures, and Languages — 7th. International Workshop, ATAL-2000, Boston, MA, USA, July 7–9, 2000, Proceedings*, Lecture Notes in Artificial Intelligence. Springer, Berlin, Berlin, 2001.
20. Peter Stone, Patrick Riley, and Manuela Veloso. Defining and using ideal teammate and opponent models. In *Proceedings of the Twelfth Annual Conference on Innovative Applications of Artificial Intelligence*, 2000.
21. Peter Stone and Manuela Veloso. Task decomposition, dynamic role assignment, and low-bandwidth communication for real-time strategic teamwork. *Artificial Intelligence*, 110(2):241–273, June 1999.
22. Peter Stone and Manuela M. Veloso. Team-partitioned, opaque-transition reinforced learning. In *RoboCup-98: Robot Soccer World Cup II*, pages 261–272, London, UK, 1999. Springer.