Inter-layer Learning Towards Emergent Cooperative Behavior

Shawn Arseneau, Wei Sun, Changpeng Zhao, Jeremy R. Cooperstock

Centre for Intelligent Machines, McGill University
3480 University St, Room 410, Montreal, QC, H3A 2A7
{arseneau | wsun | czhao | jer}@cim.mcgill.ca

Abstract

As applications for artificially intelligent agents increase in complexity we can no longer rely on clever heuristics and hand-tuned behaviors to develop their programming. Even the interaction between various components cannot be reduced to simple rules, as the complexities of realistic dynamic environments become unwieldy to characterize manually. To cope with these challenges, we propose an architecture for inter-layer learning where each layer is constructed with a higher level of complexity and control. Using RoboCup soccer as a testbed, we demonstrate the potential of this architecture for the development of effective, cooperative, multi-agent systems. At the lowest layer, individual basic skills are developed and refined in isolation through supervised and reinforcement learning techniques. The next layer uses machine learning to decide, at any point in time, *which* among a subset of the first layer tasks should be executed. This process is repeated for successive layers, thus providing higher levels of abstraction as new layers are added. The inter-layer learning architecture provides an explicit learning model for deciding individual and cooperative tactics in a dynamic environment and appears to be promising in real-time competition.

Introduction

The *real-world* as a dynamic, unpredictable environment offers numerous challenges for a researcher. Past AI techniques involving multiple agents have traditionally focused on problems in static, deterministic environments with complete information accessibility. Since more interesting problems deal with the real-world scenario, it seems only logical to test with such an environment.

To gauge whether an architecture is well-suited for an environment is difficult to accomplish, however, a common measure of performance is the time taken to finish the task. The elapsed time can be significantly reduced with the implementation of a cooperative team of agents motivated towards a common goal (Schneider-Fontan and Matarie 1998). Realizing a set of user-defined rules for inter-agent cooperation can become quite complex as all possible scenarios must be investigated. Hence machine learning has become an ideal tool for the development of multi-agent behavior. By having increasingly abstract layers of interaction between agents, a cooperative behavior emerges. Thus, multi-agent architecture in concert with machine learning techniques seems a promising path to pursue.

In order to study such real-world complexities in a limited domain the concept of RoboCup was introduced (Kitano et al. 1997, Asada et al. 1999). While maintaining an affordable problem size and research cost, RoboCup was designed in an attempt to provide a common task for the evaluation of various theories, algorithms, and architectures. This domain currently uses soccer as its "standard problem", in particular, the soccer simulator developed by Noda et al. (Noda, Matsubara and Hiraki 1998). This simulator incorporates many real-world complexities, such as limited vision, limited stamina, oral communication, and sensor noise, thus providing a convenient, yet non-trivial testbed for AI researchers.

Among the existing architectures, role-based decision trees are a common approach (Coradeschi and Karlsson 1998, Matsumoto and Nagai 1998). An agent's action is selected according to prioritized rules organized in a decision tree. However, in these works, building a decision tree is based completely on the designer's knowledge and experience, thus it would be difficult to cover all situations that might occur in a dynamic environment.

Under the influence of Brooks' work with the subsumption architecture (Brooks 1986), several layered architectures have been proposed. Matellan et al. (Matellan, Borrajo, and Fernndez 1998) used a two level structure: one composed of reactive skills capable of achieving simple actions on their own; the other based on an agenda used as an opportunistic planning mechanism to compound, activate, and coordinate the basic skills. Scerri and others (Scerri 1998; Westendorp, Scerri, and Cavedon 1998) proposed a multi-layered behavior based system, made up of a number of structurally similar layers, in which upper layers control the activation and priority of behaviors in lower layers. These layered architectures are advantageous for management and coordination, however, they have serious drawbacks due to the hard-wired design. For example, how does one set the appropriate priorities?

There is also the problem dealing with oscillations between behaviors when information indicates more than one behavior is applicable. Finally, handling uncertain information becomes yet another challenge.

In order to cope with these problems, some effort has been made to incorporate machine learning into the layered structure. Noda et al. (Noda, Matsubara and Hiraki 1996) implemented neural networks to train an agent to choose between two basic behaviors, shooting and passing. Balch (Balch 1998) used Q-learning to train individual behaviors to determine when and how to activate a particular skill, given a common set of hand-coded low-level skills. Luke et al. (Luke et al. 1998) attempted genetic programming to "evolve" an entire soccer team from a set of low-level "basic" behavior functions to be used by individual agents. These methods, however, did not fully exploit the use of machine learning for low-level behaviors.

Stone and Veloso used neural networks to learn a basic behavior, ball interception, which was then incorporated via a decision tree into the learning of passing (Stone and Veloso 1998). They further suggested an extension to incorporate learning into the *decision* between dribbling, passing, and shooting skills, though this higher level was not implemented. In a later paper, they proposed a general framework named “layered learning” to deal with intractable tasks (Stone and Veloso 1999). This approach decomposed the problem, i.e. passing the ball, into several simpler, learnable steps: ball interception, pass evaluation, and pass selection. While this method provided an innovative framework for complex task decomposition, it did not, unfortunately, provide a mechanism for the learned selection among a set of lower-layer subtasks, such as that suggested earlier (Stone and Veloso 1998).

As an extension of the behavior-based layered structure (Luke et al. 1998), a generic architecture is proposed to incorporate machine learning techniques into the decision making process for multi-agent systems. This scheme is applicable to both individual and team cooperation in a dynamic environment.

Inter-layer Learning Approach

The general layout of the proposed architecture is a tree-like structure, where each additional layer introduces a higher level of complexity and control. As an example, a three-tier structure is shown in figure 1. The team strategy layer determines for each agent which individual strategy is to be adopted. Based on the individual strategy, an agent chooses an appropriate behavior from all possible basic behaviors in the lower level. The ultimate action output of an agent is the result of a *top-down* decision making process.

Machine learning of this layered architecture is implemented in a *bottom-up* fashion. Basic behaviors are learned first. The individual strategy layer then learns to choose between basic behaviors. Finally, the team strategy layer learns to coordinate a group of learned agents.

This process could be continued as necessary to achieve the desired scale of control according to various demands of applications. For example, considering a number of agents as a single entity, an additional layer could be built to attain more complex, cooperative behaviors between these groups.

The subdivision within a specific layer is done based on the perception of the world by the agent. For example, if an agent is to drive a car, one could divide the actions of parking and passing another vehicle on the highway, as each would become a viable option only under specific environmental conditions. The subdivisions within a layer are based on certain environmental variables which may be specific to that particular layer. This presents the danger of information overload. To follow the design principle of minimalism (Werger 1999), each layer uses only information about the environment pertinent to itself to simplify the entire process (Westendorp, Scerri, and Cavedon 1998). It should also be noted that the choice of percepts and reward functions are user defined, therefore it is left for future work.

The inter-layer learning architecture is used to design a team of soccer playing agents within the RoboCup simulator (Noda et al. 1998). The soccer team employs three distinct layers: basic skills, individual strategy, and a team strategy (Figure 1). Supervised and reinforcement learning are used, although unsupervised learning is also a possible venue to explore when building the interaction between layers. Each of these layers is discussed in further details in the following sections.

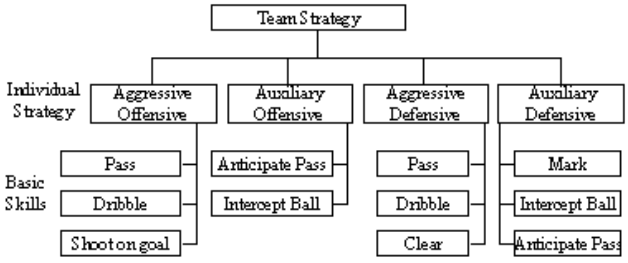


Figure 1. Overview of the inter-layer learning agent architecture. Note that the branches denote *choices* not *inputs*.

Basic Skills

A machine learning approach is employed to develop the basic skills, as other RoboCup researchers have done in the past (Stone and Veloso 1998, Ohta 1998, Tambe et al. 1998). The basic skills layer is subdivided into passing, dribbling, shooting on the goal, anticipating a pass, intercepting the ball, marking, and clearing. Neural networks are chosen for three of the skills, each based on a single action, intercept the ball, shoot on goal, and pass the ball. As it is possible to determine whether the action is appropriate soon after its execution, a supervised-learning algorithm is well suited for this type of scenario.

Dribbling and anticipating a pass are learned using

temporal difference Q-learning (Q-TD) as both of these skills involve a more complex state space and reward function. This choice is further motivated as Q-TD is an active reinforcement learning scheme that does not require the estimation of a world model, i.e. the state transition probabilities.

Both marking and clearing are analytically derived due to the nature of the skills (Stone, Veloso, and Riley 1999).

In the following discussion, we examine in further details the skills of dribbling (Q-TD), passing (neural network), and anticipating a pass (Q-TD).

Dribbling – Q-TD. The task of dribbling is not only to run with the ball towards the opponent's goal, but also to avoid opponents. Given the delayed-reward aspect of this scenario, Q-TD was chosen for this skill.

The learning agent is trained to dribble against an opponent that has been trained to intercept the ball. A state is defined by the following three variables: opponent's goal direction, opponent's distance, and opponent's direction. The agent decides among five dribbling directions. At the early stage of learning, decision-based exploration is used so that the agent chooses actions that have been picked less often. The reward is assigned to both intermediate and terminal states based on distance gained towards the opponent's goal and time consumed. The assignment of intermediate reward motivates the agent to dribble towards the opponent's goal, thus helping speed up the learning procedure.

Figure 2 shows gained distance versus epochs during the Q-TD procedure. When the maximum distance the agent is able to reach stop increasing and the minimum distance is constantly above a certain threshold, which is 20 in this case, we consider that the learning is finished. Therefore, in Figure 2, the learning ended after approximately 400 epochs.

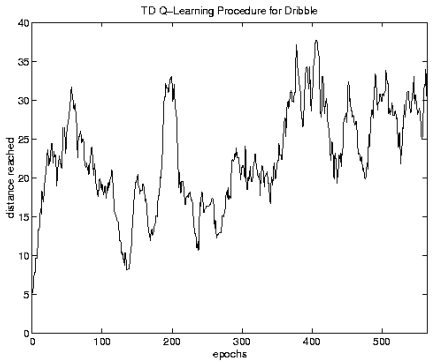


Figure 2. Q-TD procedure for dribble skill.

In a real soccer competition, the agent will use the learning result, without further Q-value updating, to dribble around the closest visible opponent. If no opponent is visible, the agent will remember the last visible opponent and behave accordingly. This learned dribble skill also has the added benefit of an emergent collision detection function as the player learns to avoid the opponents when it has possession of the ball.

Passing – BPNN. The challenge of passing the ball is in choosing the appropriate teammate to receive the pass. Since passing is not a continuous behaviour but a single action, for which active reinforcement learning is difficult to set up, the back-propagation neural network (BPNN) was chosen to learn this skill.

The passing skill is learned in a simplified environment with a passing agent, one teammate, and one opponent. Both the teammate and the opponent have learned to intercept the ball through a neural network at this point. Before learning begins, the passing agent simply passes the ball to its teammate while the opponent tries to intercept the ball. The agent's visual data is then used to train a BP neural network for the passing skill.

The input values for the neural network are the teammate's distance and direction as well as the opponent's distance and direction. The neural network converged after 292 iterations as shown in Figure 3.

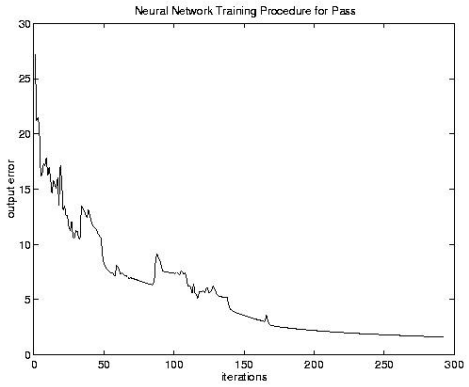


Figure 3. BP neural network training procedure for passing skill

In a real soccer competition, an agent has to choose to pass the ball to one of *m* teammates in the presence of *n* opponents, where *m* and *n* are arbitrary numbers. The trained neural network discussed above can be used as follows: First, compute the matrix *S* of probabilities of a successful pass to teammate *i* given the position of opponent *j*. Then, select teammate *i** with the highest probability of success, to receive the pass, where:

$$S_{i*} = \max_i \min_j S_{ij}.$$

This technique is similar to assigning confidence values to each of the choices (Stone and Veloso 1998).

	Success	Failure	Success rate
Random choice	254	75	77.2%
Neural network	276	53	83.9%

Table 1. Comparison of passing results by random choice and NN learning.

For evaluation, the neural network choice was compared to a random choice with three teammates and three opponents in the passing agent's field of view. The results are shown in Table 1. The successful passing rate increases by 6.7%. While the effect is marginal, it does nevertheless demonstrate improved performance.

Anticipating a Pass – Q-TD. The most difficult of the basic skills to implement is how to anticipating a pass. In a real game of soccer, a player must attempt to visualize the field from the perspective of its teammate with possession of the ball. In the RoboCup environment, the individual agents have a fixed viewing arc in which they gather visual percepts. In an attempt to learn this skill, the idea of a *viewcone* (Figure 4) is introduced in order to discretize the percepts as well as to create a simple but effective visual model along the lines of the minimalist design approach (Werger 1999).

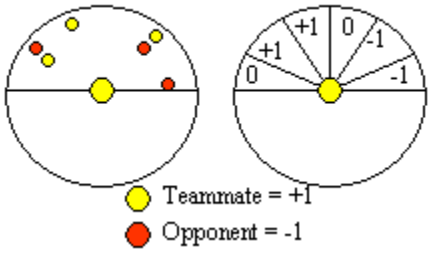


Figure 4. Viewcone of labeled arcs

The player first converts all of its information about the other players' positions on the field into global coordinates. These values are then converted to a relative coordinate system with respect to the passer. Finally, a viewcone is constructed whereby the passer parses its 90_ visual field into 15_ arcs and weights them according to the closest player. If a teammate and an opponent appear in the same arc, the closer of the two to the passer will determine the weight. These weights are entered into a Q value temporal difference learning scheme to determine the best possible action to perform. Another value used is if the passer has an unobstructed view of the player.

The four possible actions available to the player are to remain still, dash forward towards the passer, turn right and dash, or turn left and dash. In order for the player to choose the best possible actions, a unique reward system is devised. To encourage the player to dash to a location where it is open for a pass, the maximum reward is granted when the passer has an unobstructed view of the player and has two null arcs on either side (which would appear as 0, +1, 0 in the viewcone), in which case the player has no immediate threat of opponents. The reward decreases as teammates replace these null arcs, and the player is punished for being beside opponents, out of the viewcone of the passer, or within the passer's viewcone but behind another player.

After teaching the agent in a scenario with four teammates and five opponents randomly placed on one half of the field for over 28000 iterations, (with a state space of 5103 states), the player gradually chose an appropriate direction. With Q-value learning in this scenario, it is difficult to show the convergence, as the reward given after each step may produce the best action, but still results in an ineffective position. For example, if the player is outside the passer's viewcone, it may take several actions before it can achieve an improved position and hence escape from the low reward states. However, in comparing the learned skill against a random behavior, the learned skill moved in an appropriate direction approximately 70% more often.

Individual Strategy Layer

The individual strategy layer involves choosing amongst basic skills from a higher-level viewpoint (Noda et al. 1996). This layer is subdivided into four types. The first division dictates whether the agent has possession of the ball (*aggressive*) or not (*auxiliary*). This is further categorized into *offensive* and *defensive* depending on whether the agent's home area is on its own side of the field or its opponent's side (see Figure 1). The home area refers to the center of a circular region in which the agent may move. These regions are overlapped to accommodate agent interaction. The individual strategy examined here will be aggressive-defensive, which implies that the agent must choose among dribbling, passing, and clearing. As the choice to clear the ball is made analytically, when no other course of action is possible, in this instance the decision learning is between that of dribbling versus passing. This is useful for both aggressive-offensive and aggressive-defensive players.

Dribble/Pass. A learning agent is surrounded by four teammates and four opponents. All these players have been trained to dribble, pass, and intercept the ball. To simplify the learning situation, the teammate will only dribble after receiving a pass. The players pass or dribble the ball as far as possible towards the opponent's goal until an opponent successfully intercepts the ball.

The visual information of the learning agent is gathered to train a back-propagation neural network. Viewcone quantization is used here again to simplify the visual information of the learning agent. The weighted viewcone arc values discussed earlier as well as the direction of the opponent's goal are fed into the neural network.

In order to obtain the desired output of the neural network, both decisions of dribbling and passing are tested for the same configurations. The reward of each decision is calculated as:

$$\text{reward}=(\text{distance gained})-0.01\times(\text{time consumed})$$

For each configuration, the decision with the higher reward determines the desired output of the neural network.

Figure 5 illustrates the training procedure which converged after 15228 iterations to under 5% error. This behavior during our actual soccer competition proved highly effective at eluding the opposing team.

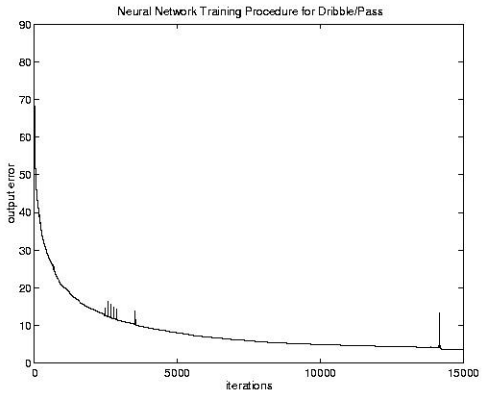
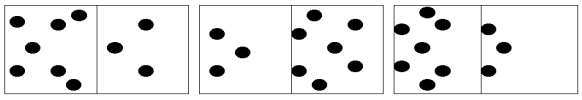


Figure 5. BP neural network training procedure for Dribble vs. Pass

Team Strategy Layer

This layer of learning investigates the performance of emergent team behavior. In order to adapt to different opponents and different scenarios, an effective team must learn to become more defensive or offensive during the match (Tambe 1996). In this particular scenario, a model-based approach to team strategy is adopted (Tambe 1996, Tambe et al. 1998) as opposed to a strictly behavior-based strategy (Werger 1999). This becomes the first layer of cooperation between the agents, and can potentially be expanded to further layers.

Three types of strategies; offensive, defensive, and half/half, are defined by three possible sets of home areas of the players (Figure 6).



(a) (b) (c)

Figure 6. Team Strategies: (a) half/half, (b) offensive, (c) defensive

To facilitate the learning of an appropriate team strategy, a *captain* agent is introduced. This individual agent would carry the burden of deciding which team strategy to adopt based on its own visual cues and relays its decision to the rest of the team. In order to account for players outside the captain's field of view, the concept of *temporal decay* is applied (Westendorp, Scerri, and Cavedon 1998).

Once the captain has received its visual information, a world-model is created. Based on this world-model, Q-TD is applied to learn the team strategy.

The positional percepts that define the state values must be discretized in order to reduce the state space to a manageable size. To maintain useful information regarding team positions and local densities of players, a field mean and pseudo variance are calculated by dividing the field into six partitions in the x-direction. These figures are then used to determine the regions corresponding to the average position of both teams. This approach is similar to the tolerance design technique (Werger 1999). For example, in Figure 7, the mean of team *A* (white) lies in region 2 and team *B* (black) in region 5. The pseudo-variance varies with the number of agents present in the mean region.

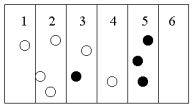


Figure 7. Discretized field

The final piece of data used by the learning algorithm is the ball location, which is also needed to create the appropriate reward. The reward relies solely on the amount the ball advanced or retreated during 100 ticks of the game clock. The choice of this time is completely heuristic.

After learning, the captain chooses between *offensive*, *defensive*, and *half/half* strategies according to the state of the game, and broadcasts this choice to the team. Each player picks a different home area corresponding to the new strategy, which determines whether the player will focus more on the opponent's side of the field (*offensive*) or on its own (*defensive*).

Results and Conclusions

Once all of the layers were integrated into a complete soccer playing agent, our team was entered into the 1999 RoboCup competition at McGill University, outscoring the competition by a total of 12:0. The inter-layer learning architecture appears to be a reasonable structure for a multi-agent, cooperative task. Employing the agents to learn cooperative behaviors, as opposed to taking an analytical approach, allowed emergent team strategies to be adopted.

It should be noted that although the final behavior is emergent, the human designer is still responsible for the division of the sub-tasks as well as the choice of training data. However, this approach extends the application of machine learning from the acquisition of individual tasks to the learning of the appropriate selection criteria for choosing between them. Expanding this architecture to cooperation between sub-teams working collectively appears possible, and an implementation of such a structure is left for further research.

Acknowledgements

We would like to thank Sorin Lerner for all of his help with the RoboCup soccer server, without whom we would never have gotten our players to see, let alone kick the ball.

References

Asada, M., Kitano, H., Noda, I., and Veloso, M. 1999. RoboCup, Today and Tomorrow – What We have Learned. *Artificial Intelligence*. 110(2):193-214.

Balch, T. 1998. Integrating Learning with Motor Schema-Based Control for a Robot Soccer Team. *RoboCup-97: Robot Soccer. World Cup I*. 483-491. Berlin, Germany: Springer-Verlag.

Brooks, R. A. 1986. A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14-23.

Coradeschi, S., and Karlsson, L. 1998. A role-based decision-mechanism for teams of reactive and coordinating agents. *RoboCup-97: Robot Soccer. World Cup I*. 99-111. Berlin, Germany: Springer-Verlag.

Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E., and Matsubara, H. 1997. RoboCup: A Challenge Problem of AI. *AI Magazine*. 18:73-85.

Luke, S., Hohn, C., Farris, J., Jackson, G., and Hendler, J. 1998. Co-evolving Soccer Softbot Team Coordination with Genetic Porgramming. *RoboCup-97: Robot Soccer. World Cup I*. 398-411. Berlin, Germany: Springer-Verlag.

Matellan, V., Borrajo, D., and Fernndez, C. 1998. Using ABC^2 in the RoboCup domain. *RoboCup-97: Robot Soccer. World Cup I*. 475-482. Berlin, Germany: Springer-Verlag.

Matsumoto, A., and Nagai, H. 1998. Decision making by the characteristics and the interaction in multi-agent robotics soccer. *RoboCup-97: Robot Soccer. World Cup I*. 132-143. Berlin, Germany: Springer-Verlag.

Noda, I., Matsubara, H. and Hiraki, K. 1996. Learning Cooperative Behavior in Multi-Agent Environment – A Case Study of Choice of Play-plans in Soccer. *Proceedings of the 4th Pacific Rim International Conference on Artificial Intelligence*. 570-579.

Noda, I., Matsubara, H., Hiraki, K., and Frank, I. 1998. Soccer Server: A Tool for Research on Multiagent Systems. *Applied Artificial Intelligence* 12(2-3):233-250.

Ohta, M. 1998. Learning Cooperative Behaviors in RoboCup Agents. *RoboCup-97: Robot Soccer. World Cup I*. 412-419. Berlin, Germany: Springer-Verlag.

Scerri, P. 1998. A Multi-layered Behavior-Based System for Controlling RoboCup Agents. *RoboCup-97: Robot Soccer. World Cup I*. 467-474. Berlin, Germany: Springer-Verlag.

Schneider-Fontan, M. and Matarie, M. 1998. Territorial Multi-Robot Task Division. *IEEE Transactions on Robotics and Automation*. 14(5):815-822.

Stone, P. and Veloso, M. 1998. A Layered Approach to Learning Client Behaviors in the RoboCup Soccer Server. *Applied Artificial Intelligence* 12:165-188.

Stone, P. and Veloso, M. 1999. Layered Learning. *International Joint Conference on Artificial Intelligence Workshop on Learning About, From, and With Other Agents.*

Stone, P., Veloso, M. and Riley, P. 1999. The CMUnited-98 Champion Simulator Team. *RoboCup-98: Robot Soccer. World Cup II*. Berlin, Germany: Springer-Verlag. At URL: http://www.cs.cmu.edu/afs/cs/usr/pstone/public/papers/98springer/final-champ/final-champ.html.

Tambe, M. 1996. Tracking Dynamic Team Activity. In Proceedings of the thirteenth Conference on Artificial Intelligence Applications. 11:80-87. Cambridge, U.S.A.: MIT Press.

Tambe, M., Adibi, J., Al-Onaizan, Y., Erdem, A., Kaminka, G., Marsella, C., and Muslea, I. 1998. Building Agent Teams Using an Explicit Teamwork Model and Learning. *Artificial Intelligence*. 110(2):215-239.

Werger, B. 1999. Cooperation Without Deliberation: A Minimal Behavior-Based Approach to Multi-Robot Teams. *Artificial Intelligence*. 110(2):293-320.

Westendorp, J., Scerri P., and Cavedon L. 1998. Strategic Behaviour-Based Reasoning with Dynamic, Partial Information. *RoboCup-97: Robot Soccer. World Cup I*. 297-308. Berlin, Germany: Springer-Verlag.