Automatic Recognition of Human Team Behaviors

Gita Sukthankar and **Katia Sycara**
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
{gitars,katia+}@cs.cmu.edu

Abstract

This paper describes a methodology for recording, representing, and recognizing team behaviors performed by human players in an Unreal Tournament MOUT (Military Operations in Urban Terrain) scenario. To directly monitor the performance of human players, we developed a customized version of Unreal Tournament that records position and orientation of all the team members through time as they participate in a simulated MOUT scenario of a firing team moving through an urban area. Behavior recognition is performed offline using a set of hidden Markov models on short movement sequences that are translated into a canonical reference frame; the behavior model with the highest log likelihood for a given sequence is identified as correct. We believe that accurate offline recognition of team behaviors is an important prerequisite towards building virtual training environments for teamwork tasks.

1 Introduction

In certain domains tasks are too complicated to be performed by individual agents and must be achieved through the coordinated efforts of a group of agents over a period of time. To analyze performance of these tasks, we need to extend existing behavior recognition formalisms to accommodate group behaviors. In multi-agent systems we can often eliminate the need for behavior recognition by having the agents communicate their states, intentions, and plans; however when the group of agents includes human participants it becomes unreasonable to demand exhaustive self-reporting. Also, in adversarial domains where disguising ones future plans confers strategic advantage, good plan recognition allows the accurate analysis of groups of agents who are unwilling to report their state.

In this paper we examine the problem of team behavior recognition in physical domains. We make the assumption that the actions of the agents are tightly coupled; all agents are always performing tasks relevant to accomplishing team goals and the team is never concurrently executing multiple group behaviors. Also, we focus on physical domains in which the majority of the agent's actions involve movement. Due to the increase in number of actions generated, assuming that each team member is simultaneously executing actions, team behaviors have a more complicated temporal structure than single agent behaviors. However group plans involving physical movement possess a spatial structure which can be exploited to classify team behaviors. This spatial structure includes the relative physical position of team members and the physical position of team members in relation to static landmarks (e.g., walls and doorways).

One aspect of our domain which makes the problem more challenging is that our teams are composed of human players executing actions by moving bots in a simulated environment; the humans typically exhibit more variability in their movements than programmed bots who path plan the most efficient trajectory towards their next target. This additional variability makes the recognition problem more difficult; also data collection is an expensive process since it requires multiple humans to participate in the scenario.

This paper describes a methodology for recording, representing, and recognizing team behaviors performed by human players in an Unreal Tournament MOUT (Military Operations in Urban Terrain) scenario. To directly monitor the performance of human players, we developed a customized version of Unreal Tournament that records position and orientation of all the team members through time as they participate in a simulated MOUT scenario of a firing team moving through an urban area. Although we have only examined the MOUT domain, we believe that this methodology could be extended to analyze other military and sports tasks that utilize tightly coupled teamwork behaviors in physical domains.

2 Related Work

Related work on the problem of human behavior recognition has emerged from three communities: computer vision researchers who have examined the problem of activity inferencing with temporal constraints [Shi *et al.*, 2004], graphics researchers who address the problems of clustering and segmenting unlabeled motion capture data [Jenkins and Matarić, 2002; Barbic *et al.*, 2004] and artificial intelligence plan recognition researchers [Allen *et al.*, 1991; Charniak and Goldman, 1993; Huber *et al.*, 1994; Tambe and Rosenbloom, 1995; Pynadath, 1999; Goldman *et al.*, 1999; Bui, 2003] who have traditionally focused on inference mechanisms.

Human behavior recognition is also performed in tutoring systems, such as the one developed for the MOUT domain by [Livak, 2004]. In model-tracing tutoring systems the same cognitive model is used to generate human-like behavior of computer generated forces and as part of the recognition system used to evaluate the human trainee's performance. In our system we focus on analyzing and evaluating the complex spatial relationships that exist between the human team members rather than using simple location descriptors such as 'in room' to evaluate correctness of behavior.

There has also been work on the problem of identifying team behaviors from snapshots of formations in Robocup [Riley and Veloso, 2000; Riley *et al.*, 2002] and in MOUT [Sukthankar and Sycara, 2005]. This is very useful for classifying certain team behaviors that include highly discriminative static spatial relationships that rarely appear in other behaviors; however certain behaviors (e.g., bounding overwatch) are only distinguishable by examining sequences of spatial relationships as they evolve through time. In this paper, we use both spatial and temporal characteristics to perform behavior recognition by using invariant spatial relationships in our state representation and temporal relationships in the transition matrix of our hidden Markov models. Hidden Markov models have been successfully applied to the problem of robotic behavior recognition by [Han and Veloso, 1999].

3 MOUT Domain

MOUT (Military Operations in Urban Terrain) scenarios involve moving teams of soldiers and vehicles through heavily cluttered urban areas to accomplish high-level strategic objectives. Typically small teams (4 firing-team pairs) of soldiers work to achieve their objectives in a tightly coupled fashion without external guidance from central command.

In our MOUT scenario, we focus on the building clearing task, during which a firing-team of soldiers must enter a building and clear it of enemy occupants. The cognitive task analysis of building clearing given in [Phillips *et al.*, 2001] identifies the process of approaching the building as one of the key decision requirements that dictates whether the process is successful; this paper focuses on the three behaviors, stacked movement, bounding overwatch, and buttonhook entry used to approach and enter a building. These behaviors are difficult to identify solely on the basis of static snapshots due to their spatial similarity.

During stacked movement the purpose is to move the team in such a way that their gun angles completely span all possible areas of approach; the team moves slowly and in synchronization. For moving through open areas or intersections, this approach is less feasible since it's hard to span all possible threatened areas. In this case, the bounding overwatch behavior is used; one soldier moves forward while the other remains stationary. The buttonhook entry is similar to bounding overwatch; one soldier moves through the doorway hugging the wall while the other soldier waits and guards. After the entry is clear, the second soldier moves through the doorway hugging the opposite wall.

Figure 1: MOUT scenario in customized Unreal Tournament environment from spectator viewpoint. A pair of human players control soldiers A and B as they move through a small urban layout. The bot models and animations were modified to conform to the appearance of real human soldiers rather than the larger-than-life UT fantasy fighter models.

4 Method

The results described in this paper were obtained using the following procedure:

1. Pairs of human players using a modified Unreal Tournament game interface manipulated "bots" through a small urban layout.
2. After some initial practice familiarizing themselves with the map and working together as a team, the subjects were instructed to perform a particular sequence of team behaviors. Note that the subjects were not playing an Unreal Tournament level, but using Unreal Tournament to execute sequences of commonly used MOUT team maneuvers.
3. Using the modified version of Unreal Tournament described in Section 4.1, traces of the players' behaviors were recorded in a text file for offline analysis.
4. Behavior traces were preprocessed to generate short, overlapping segments and converted into a canonical reference frame based on the motion of the team's centroid as described in Section 4.2.
5. Offline traces were automatically classified using the set of hidden Markov models as described in Section 4.3.
6. The results of our automatic recognition were compared with a manually annotated version of the trace.

Section 5 gives preliminary results obtained with a single pair of players who performed the task multiple times in different layouts.

4.1 Data Collection

To directly monitor the performance of human players, we customized the first-person shooter game, Unreal Tournament

(UT), using the game development language Unrealscript. Many of the original UT game classes were written in Unrealscript and thus can be directly subclassed to produce modified versions of the game (known as mods); for example, Gamebots [Kaminka *et al.*, 2002] is an example of a mod that allows external programs to control game characters using network sockets.

We developed our own `TrainingBot` mod that allows us to save the state of all the bots in the scenario; currently we save each player's ID number, position $(x, y.z)$, and rotation (θ,ϕ) every 0.15 seconds. This information is useful for both offline behavior analysis and for a separate replay mode that allows us to create bots that follow the paths recorded by the original players. To ensure that the player traces are correctly synchronized in time, we wait until all players have activated their autosave option to begin recording traces.

To enhance the immersive experience of the players, we created a custom MOUT soldier model (see Figure 1) that conforms more closely to the appearance of a real soldier than the standard UT fantasy fighter models. We also developed a set of hand-signal animations to allow the players to use a small set of military hand signals triggered on key-presses. For our experiments, players executed team behaviors in an uncluttered map environment of corridors and rooms; in the future we plan to extend the scenario to include less structured regions with more clutter.

4.2 Representation

Due to the continuous nature of the domain, automatically determining the exact transition points between team behaviors is a difficult problem. While approaching and entering buildings, the players continue moving their bots, changing team behaviors as appropriate for the physical layout. We address this issue by dividing the traces into short, overlapping time windows during which we assume that a single behavior is dominant; these windows are classified independently as described in Section 4.3. To recognize team behaviors performed in different physical layouts, it is important for our classifier to be rotationally and translationally invariant; we achieve this by transforming the data in each window into a canonical coordinate frame as described below.

More formally, we define:

- $a \in 1, \ldots, A$ is an index over A agents;
- j is an index over W overlapping windows;
- $t \in 1, \ldots, T$ is an index over the T frames in a given window;
- $\mathbf{x}_{a,j,t}$ is the vector containing the (x, y) position of agent a at frame t in window j.

The centroid of the positions of the agents in any given frame can be calculated as:

$$\mathbf{C}_{j,t} = \frac{1}{A} \sum_{\forall a} \mathbf{x}_{a,j,t}.$$

We describe the configuration of the agent team at any given time relative to this centroid to achieve translation invariance. However, rather than rotating each frame independently we define a shared canonical orientation for all the frames in a window. This is important because it allows us to distinguish between similar formations moving in different directions (e.g., agents moving line abreast vs. single file). One standard technique for defining a canonical orientation is to use the principal axis of the data points for that window, which can be calculated using principal component analysis (PCA). However for efficiency we have empirically determined that we can achieve similar results by defining the canonical orientation as the displacement of the team centroid over the window: $\mathbf{d}_j = \mathbf{C}_{j,T} - \mathbf{C}_{j,1}$.

We rotate all of the data in each window so as to align its canonical orientation with the x-axis, using the rotation matrix $\mathbf{R}_j$. Thus the canonical coordinates, $\mathbf{x}'$, can be calculated as follows: $\mathbf{x}'_{a,j,t} \equiv \mathbf{R}_j \mathbf{x}_{a,j,t} - \mathbf{c}_{j,t}$. Our current recognition technique (described in Section 4.3) also relies on observations of agents' velocity as a feature which we locally compute as: $\mathbf{v}_{a,j,t} \equiv ||x'_{a,j,t+1} - x'_{a,j,t}||$.

4.3 Classification

For each canonically transformed window in our trace, our goal is to select the best behavior model. We perform this classification task by developing a set of hidden Markov models (HMMs), one for each behavior b, and selecting the model with the highest log-likelihood of generating the observed data. Our models ($\{\lambda_b\}$) are parameterized by the following:

- N, the number of hidden states for the behavior;
- $\mathbf{A} = \{a_{ij}\}$, the matrix of state transition probabilities, where $a_{ij} = Pr(q_{t+1} = j | q_t = i), \forall i, j$ and q_t denotes the state at frame t;
- $\mathbf{B} = \{b_i(o_t)\}$, where $b_i(o_t) = \mathcal{N}(\mu_i, \Sigma_i)$. The observation space is continuous and approximated by a single multivariate Gaussian distribution with mean, μ_i and a covariance matrix, Σ_i, for each state i.
- $\pi = \{\pi_i\}$, the initial state distribution.

For our problem, given A agents in a team, the observations at time t and window w are the tuple: $o_t = (\mathbf{x}'_{1,w,t}, v_{1,w,t}, \ldots, \mathbf{x}'_{A,w,t}, v_{A,w,t})$. We determine the structure for each behavior HMM based on our domain knowledge. For instance, the stacked behavior can be described using only two states ($N = 2$), whereas we represent the more complicated bounding overwatch behavior using six states connected in a ring. Each hidden state captures an idealized snapshot of the team formation at some point in time, where the observation tuple (in canonical coordinates) is well modeled by a single Gaussian. Rather than initializing the HMMs with random parameters, we use reasonable starting values. These can be polished using expectation-maximization (EM) [Duda *et al.*, 2001] on labeled training data.

To determine the probability, $Pr(o_{1\ldots T}|\lambda_b)$, of generating the observed data with the model λ_b, we employ the forward algorithm [Rabiner, 1989] as implemented in the Hidden Markov Model toolbox [Murphy, 2001]. We classify each window segment with the label of the model that generated the highest log-likelihood.

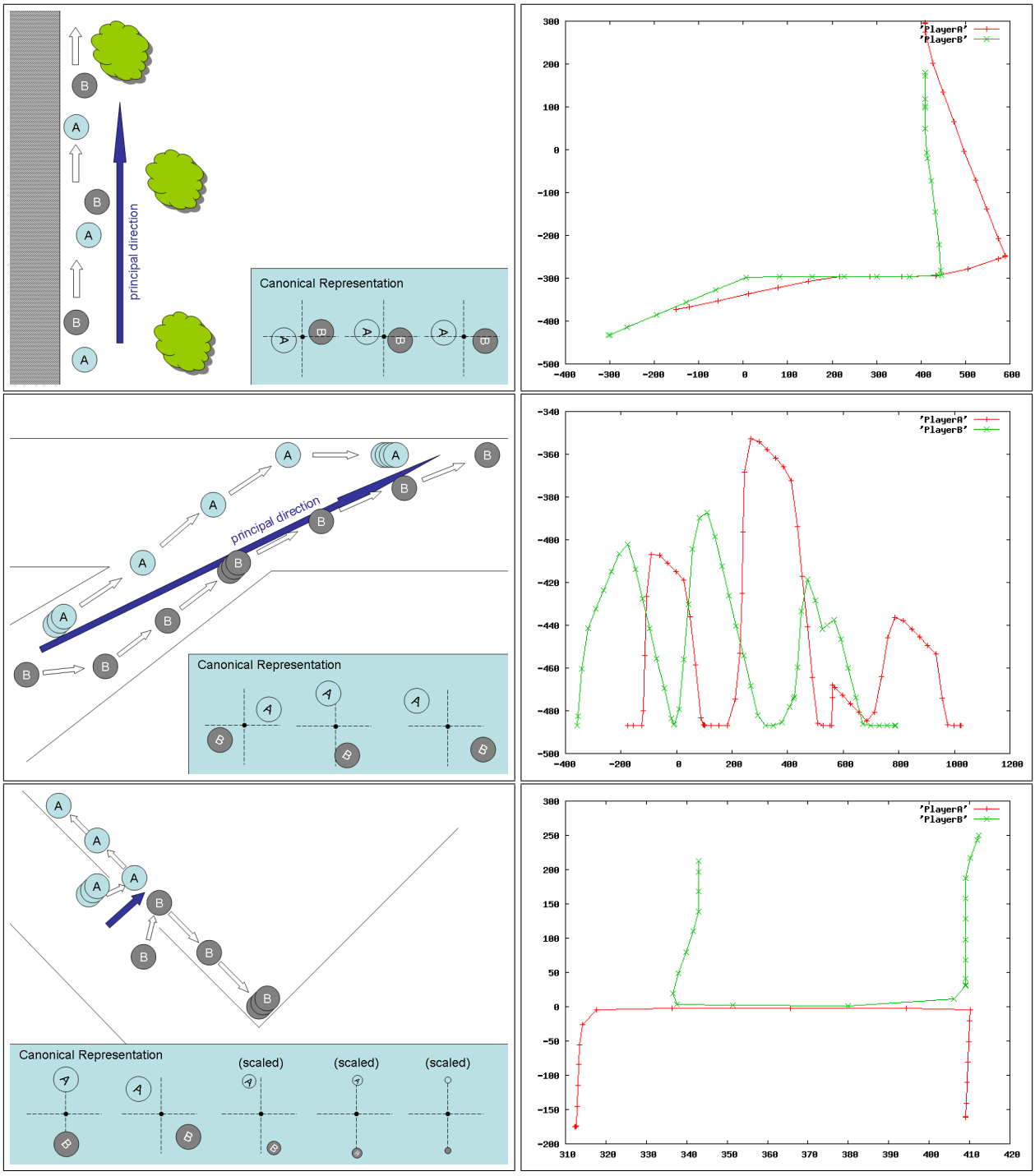

Figure 2: Team Behaviors: Stacked Formation (top), Bounding Overwatch (middle), Buttonhook Entry (bottom). Schematics for each behavior, along with the canonical representation for several frames, are depicted in the left column. A sample raw trace for each behavior is shown in the right column; the coordinates of the axes are in Unreal Tournament length units.

Table 1: Confusion matrix for HMM behavior classification. The ground truth is given in the left column; the classification result is given in the top row. Our hidden Markov model approach achieves good accuracy. Buttonhook entry is often confused with bounding overwatch, as may be expected from similarities in the canonical representation as shown in Figure 2.

	stacked	bounding	buttonhook
stacked	90%	10%	0
bounding	14%	67%	19%
buttonhook	0%	33%	67%

5 Results

To evaluate our automatic recognition method, we developed behavior models for the standard team behaviors used by a 2-person firing team during the approach phase of the MOUT building clearing task: stacked movement, bounding overwatch, and buttonhook entry (see Figure 2). A pair of human players performed sequences of team behaviors in our Unreal Tournament urban simulation; position data was recorded from both of the players at 0.15 second intervals using our `TrainingBot` mod (as described in Section 4.1). Players executed team behaviors in predesignated sequences, transitioning instantly from one behavior into the next, adapting each behavior as needed to the local physical layout (turning corridors, entering rooms). Figure 2 (right) shows a raw trace for each behavior; note that even consecutive executions of the same behavior exhibit significant variation as shown by the bounding overwatch behavior. Player traces were divided into overlapping 20 frame (3 second) windows, which were converted into a canonical coordinate frame as described in Section 4.2. This process is illustrated in the inset of Figure 2. The window size was empirically determined by observing the average speed of the players' maneuvers.

Table 1 presents the classification results (confusion matrix) for the three modeled behaviors; the accuracy of the HMM approach is good, particularly for the stacked formation. Buttonhook entry is sometimes confused with bounding overwatch, as may be expected from similarities in the canonical representation as shown in Figure 2. In Section 6 we discuss ways that domain knowledge could be incorporated to improve these recognition results.

6 Discussion

The experiments described in the previous section were deliberately designed to omit many spatial and temporal cues that would exist in a real MOUT scenario. This enables us to examine the raw accuracy of our classifier. For instance, we chose not to incorporate the existence or position of static spatial features (e.g., doors and walls) into our observation model, even though this impacts the team's choice of behavior in a more realistic MOUT task. For example, the buttonhook entry behavior is usually performed at doorways or windows whereas stacked movement typically occurs when the team is paralleling a wall.

In our experiments, the sequence of behaviors was arbitrarily chosen to create a variety of behavior transition opportunities. Our HMMs perform recognition at a very low-level and do not currently exploit inter-window dependencies. Realistically, some behaviors more commonly occur before others; for example, in a typical building clearing operation, there are often long periods of bounding overwatch followed by a single buttonhook entry through a doorway, followed by another period of bounding overwatch.

By exploiting higher-level domain knowledge about the building clearing task, we believe that we can improve recognition performance over a full-length MOUT scenario in which the subjects are instructed to perform team behaviors as appropriate for the situation. This could be done using a hierarchical approach, such a hierarchical HMM [Fine *et al.*, 1998] that examines longer time sequences at multiple resolutions. We propose the following strategy for automatic recognition of a full MOUT scenario:

1. Annotate the map (either manually or automatically) with relevant features, such as doors, intersections, windows, and trees.

2. Employ the HMMs described above to perform local analysis and assign discrete labels to each window.

3. Use these as discrete observations for higher-level HMMs that examine longer time windows and incorporate map annotations into the observation space.

A simpler approach is to assign a prior probability to each low-level HMM based on the current map context and select the behavior that maximizes the *a posteriori* probability. We plan to test these approaches in future experiments over a full-length MOUT scenario.

7 Conclusion

This paper describes a methodology for recording, representing, and recognizing team behaviors performed by human players in an Unreal Tournament MOUT scenario. Simulation traces of the players' positions were preprocessed to generate short, overlapping segments and converted into a canonical reference frame based on the motion of the team's centroid. Segments were automatically classified using a set of hidden Markov models and selecting the model with the highest log-likelihood. We propose a method for extending our technique to perform behavior recognition on a full-length MOUT scenario using a hierarchical approach and exploiting domain knowledge.

For future work, we plan to evaluate our recognition method on a larger pool of human subjects to ensure that our technique adequately handles behavior execution variability between subjects. We are particularly interested in the problem of modeling subject error since subjects occasionally execute the wrong action or poorly execute the correct behavior. Also, we intend to expand our team size to include multiple, simultaneous, firing-teams and to expand our behavior model library accordingly.

Acknowledgments

The authors thank Reid Van Lehn and Jason Yates for their development and modeling contributions to our Unreal Tournament experimental software. Rahul Sukthankar reviewed this document and provided helpful feedback.

References

[Allen *et al.*, 1991] J. Allen, H. Kautz, R. Pelavin, and J. Tennenberg. *Reasoning About Plans*, chapter 2, pages pp.121–148. Morgan Kaufmann Publishers, 1991.

[Barbic *et al.*, 2004] J. Barbic, A. Safonova, J-Y. Pan, C. Faloutsos, J. Hodgins, and N. Pollard. Segmenting motion capture data into distinct behaviors. In *Proceedings of Graphics Interface 2004 (GI'04)*, 2004.

[Bui, 2003] H. Bui. A general model for online probabilstic plan recognition. In *Proceedings of the International Conference on Artificial Intelligence (IJCAI)*, 2003.

[Charniak and Goldman, 1993] E. Charniak and R. Goldman. A Bayesian model of plan recognition. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 1993.

[Duda *et al.*, 2001] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley & Sons, Inc, 2001.

[Fine *et al.*, 1998] S. Fine, Y. Singer, and N. Tishby. Hierarchical Hidden Markov Model: Analysis and applications. *Machine Learning*, 32(1), 1998.

[Goldman *et al.*, 1999] R. Goldman, C. Geib, and C. Miller. A new model of plan recognition. In *Proceedings of UAI-1999*, 1999.

[Han and Veloso, 1999] K. Han and M. Veloso. Automated robot behavior recognition applied to robotic soccer. In *Proceedings of the IJCAI-99 Workshop on Team Behavior and Plan Recognition*, 1999.

[Huber *et al.*, 1994] M. Huber, E. Durfee, and M. Wellman. The automated mapping of plans for plan recognition. In *Procceedings of Uncertainty in Artifcial Intelligence (UAI)*, 1994.

[Jenkins and Matarić, 2002] O. Jenkins and M. Matarić. Deriving action and behavior primitives from human motion data. In *Proceedings of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-2002)*, 2002.

[Kaminka *et al.*, 2002] G. Kaminka, M. Veloso, S. Schaffer, C. Sollitto, R. Adobbati, A. Marshall, A. Scholer, and S. Tejada. Gamebots: A flexible test bed for multiagent team research. *Communications of the ACM*, 45(1), 2002.

[Livak, 2004] T. Livak. Collaborative warrior tutoring. Master's thesis, Worcester Polytechnic Institute, 2004.

[Murphy, 2001] K. Murphy. Bayes Net Toolbox for Matlab. *Computing Science and Statistics*, 33, 2001.

[Phillips *et al.*, 2001] J. Phillips, M. McCloskey, P. McDermott, S. Wiggins, and D. Battaglia. Decision-centered MOUT training for small unit leaders. Technical Report 1776, U.S. Army Research Institute for Behavioral and Social Sciences, 2001.

[Pynadath, 1999] D. Pynadath. *Probabilistic Grammars for Plan Recognition*. PhD thesis, University of Michigan, 1999.

[Rabiner, 1989] L. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77, 1989.

[Riley and Veloso, 2000] Patrick Riley and Manuela Veloso. On behavior classification in adversarial environments. In Lynne E. Parker, George Bekey, and Jacob Barhen, editors, *Distributed Autonomous Robotic Systems 4*, pages 371–380. Springer-Verlag, 2000.

[Riley *et al.*, 2002] Patrick Riley, Manuela Veloso, and Gal Kaminka. An empirical study of coaching. In H. Asama, T. Arai, T. Fukuda, and T. Hasegawa, editors, *Distributed Autonomous Robotic Systems 5*, pages 215–224. Springer-Verlag, 2002.

[Shi *et al.*, 2004] Y. Shi, Y. Huang, D. Minen, A. Bobick, and I Essa. Propagation networks for recognizing partially ordered sequential action. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2004)*, 2004.

[Sukthankar and Sycara, 2005] Gita Sukthankar and Katia Sycara. Identifying physical team behaviors from spatial relationships. In *Proceedings of Behavior Representation in Modeling and Simulation Conference (BRIMS)*, 2005.

[Tambe and Rosenbloom, 1995] M. Tambe and P. Rosenbloom. RESC: An approach to agent tracking in a real-time dynamic environment. In *Proceedings of the International Conference on Artificial Intelligence (IJCAI)*, 1995.