A system for monitoring and interpreting team actions of embodied agents

Brandyn White and Ladislau Bölöni
School of Electrical Engineering and Computer Science
University of Central Florida
Orlando, FL 32816–2450
Email: bwhite,lboloni@eecs.ucf.edu

Introduction

Understanding and recognizing team actions executed by embodied agents (e.g., robots, agents, humans) has applications in team sports [2], surveillance [1], and training. Many team actions can be described by the individual team member's movement patterns; moreover, by understanding the internal structure of the team action, we can predict the next actions of the individual members, as well as whether they are executing the action correctly or not.

In this demonstration we show a system which, starting from a video recording of a team action performed by humans or robotic agents, can recognize the team action, identify the teams and subteams, and determine the current state of the team action. This information allows the prediction of the future path of the team members, and their adherence to the canonical model of the action. The results of the analysis and prediction are displayed as an overlay over the video sequence (see Figure 1).

Our demonstration will exhibit the system on a collection of datasets which exemplify the bounding overwatch military maneuver. Circumstances permitting, we will also demonstrate through video recordings acquired on location at the conference.

As a note, our current system can not perform fully real-time processing of the video input - some manual post-processing being necessary. We expect that by the time of the conference we will have a system which can perform real-time processing provided that the video recording is performed in an uncluttered environment to reduce the tracking complexity.

In the following we describe the various components of the workflow: data collection, team action representation, team action analysis, and visualization.

Data acquisition

The data acquision step is similar to what would happen in practice in a surveillance application or training monitoring application. A scene in which several agents are active is recorded with a video camera. Information concerning the location, movement and heading is extracted; in our current workflow they are provided by manual annotation. The image coordinates are transformed to world coordinates using image to world homographies and the resulting output data is represented in the Viper GT format commonly used by object detection and tracking algorithms together with the original image files.

Cite as: A System for Monitoring and Interpreting Team Actions of Embodied Agents, Brandyn White, Ladislau Bölöni, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1421 – 1422

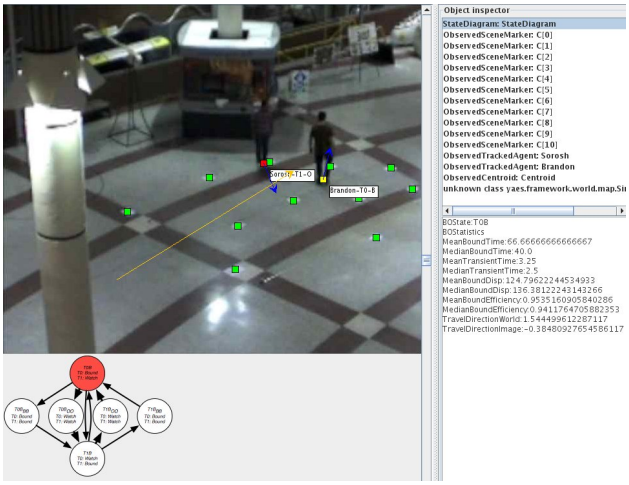

Figure 1: Screenshot of the analysis engine visualization showing the source image overlaid with agent, team, and overall Bounding Overwatch information.

Representing team actions

We are considering team actions which can be represented as a specific movement pattern of the team members. Many military maneuvers, the positioning of bodyguards around a VIP, and team sport formations fall in this category. For our demonstration, we have chosen the bounding overwatch military maneuver, which requires two subteams to alternate advancing or protecting the advancing team.

Bounding Overwatch is relatively difficult to analyze because it requires a sequence of observations to identify. There are two principal states, bounding and overwatching can be identified only over a longer series of observations. This ideal model, however, fails to properly model the transient behavior observed in reality where both teams may be performing the same role for a short period of time. To model the transient cases it is necessary to add two states after each bounding state: both teams bounding and both teams overwatching.

We represent the bounding overwatch with a state machine where each state encapsulates the current state of *both* teams (see Figure 2). The state is labeled based on the team that has been bounding last if neither is bounding or longest if at least one team is (e.g., T0B implies T_0 is bounding and T_1 is overwatching). The possible team roles are discussed below:

B The team is bounding and their motion is directed toward a

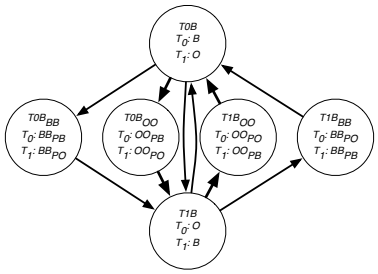

Figure 2: State diagram showing primary state transitions for both participating teams and individual roles within those states. All states have recurrent links which are omitted for clarity.

cover position located within protective cover of the overwatching team. Their direction of travel advances the team closer to the final destination.

O The team is overwatching, they are stationary, located at a cover position, and are guarding the bounding team.

$\mathbf{OO_{PB}}$ The team has finished bounding and is preparing to guard the other team as they start to bound.

$\mathbf{OO_{PO}}$ The team is preparing to bound as the other team has just finished bounding.

$\mathbf{BB_{PB}}$ The team is finishing bounding while the other team has started bounding.

$\mathbf{BB_{PO}}$ The team has started bounding as the other team is finishing bounding.

Analyzing team actions

The next step of our workflow is to analyze the input data, recognize the teams (including their subdivisions in subteams) and recognize the team action being performed. Furthermore, we need to identify the current state of the team action execution. Many team actions have alternative ways of execution, we also need to identify which of these were chosen. Once this information is available, we can use it to predict the individual agent's next action (under the assumption that they are executing the team action correctly). If the actions of the agents differ from the predicted ones, it can be interpreted as an incorrect execution. This information can be exploited by a trainer to give feedback to the trainees or it can be exploited by an adversary in a military or homeland security setting.

Let us now see how this analysis is performed in our chosen example of the bounding overwatch team action.

The first step is to identify the two subteams of the action. This step is accomplished by clustering using Lloyd's algorithm - the chosen clustering is the one that minimizes the overall K-Means error sum over the image sequences. Then, we are extracting a number of discrete features over the subteams. These features will be used to discriminate between the states specified in Figure 2. The team features used are whether the team is traveling, near cover, or watching another team (i.e., visually tracking it).

The detection of the current state is done by building a Dynamic Bayesian Network (DBN) which describes the transitions between the states in the time domain. The DBN used to represent the bounding overwatch consists of three evidence nodes: one for each team's features and a transient state timer. The transient state timer fires whenever the same state has been occupied for a specified duration of time in order to place an upper bound on the transient state durations which should naturally be short; consequently, this addition dramatically improves the discriminating power of the analyzer when considering several different team actions. Both teams share the same sensor model due to symmetry between roles. A Sampling Importance Resampling (SIR) Particle Filtering algorithm is used to perform inference on the hidden state for each time-step. At the same time, we are computing the overall probability of Bounding Overwatch, which allows us to discriminate between different team actions, as well as identify incorrectly executed team actions.

Demonstration

The GUI (see Figure 1) takes in the estimated states and the overall bounding overwatch probabilities from the Particle Filtering algorithm along with the source images, agent coordinates, agent headings, and team clusterings. The images are displayed with the agent level information overlaid graphically (e.g., position, heading, name, team) along with higher level information such as overall travel direction, team role, and overall Bounding Overwatch state. The agent positions are displayed on the current image as a square colored yellow for T0 and red for T1, areas representing protective cover (e.g., trees, buildings) are displayed as green squares, and the agent headings are displayed as blue arrows. The overall travel direction is computed as the angle from the initial team centroid location to the current team centroid location for both teams shown as an orange arrow. This assumes that the overall travel direction is linear; however, due to the common application of Bounding Overwatch to advance in hostile territory it would be highly irregular to reverse directions and though the travel is often non-linear locally in time, it is often linear overall. The current state information and agent roles are show below the video in the form of Figure 2 with the active overall state displayed in red.

There are several useful statistics that are computed including mean/median bounding duration/distance and mean/median transient (i.e., between bounding and overwatching) duration. To find the median efficiency of the bounding overwatch execution we take the median time bounding divided by the median time between bounds (i.e., bounding time added to transient time). The distance covered between bounds can be used to determine if the teams are moving too far away to effectively protect each other or if they are moving in such small bounds that little ground is covered. As Bounding Overwatch is used when traveling it is desirable to spend the majority of the time traveling or guarding and not in the transient state in between where neither team is making progress or when both teams are moving and neither team is properly guarded.

The proposed analysis engine and visualization is immediately useful for training, planning, and adversarial purposes by conveniently displaying high level information inferred from the low level coordinate information. While we used this approach to evaluate Bounding Overwatch action analysis, this same framework can be adapted to other team actions by developing a relevant DBN for the action of interest and computing input features from which the hidden state of the DBN can be inferred. [1]

REFERENCES

[1] S. Hongeng, R. Nevatia, and F. Bremond. Video-based event recognition: activity representation and probabilistic recognition methods. *Computer Vision and Image Understanding*, 96(2):129–162, 2004.

[2] S. S. Intille and A. Bobick. Recognizing planned, multi-person action. *Computer Vision and Image Understanding*, 81(3):414–445, 2001.

[1]This work was partially funded by NSF Information and Intelligent Systems division under award 0712869.