16

Effective Multi-Model Motion Tracking Under Multiple Team Member Actuators

Yang Gu and Manuela Veloso
Computer Science Department, Carnegie Mellon University
USA

1. Introduction

Autonomous robot agents need to be able to track moving objects. When tracking is performed by a robot executing specific tasks acting over the target being tracked, such as a Segway RMP soccer robot grabbing and kicking a ball, the motion model of the target becomes dependent on the robot's actions. The robot's tactic provides valuable information in terms of the target behavior. We introduced a play-based motion modeling and tracking in such scenarios (Gu & Veloso, 2006). Observations from the sensors might consist of multiple measurements due to the moving objects and the clutter. Generally the clutter has similar color as the targets we are interested in and it causes multiple hypotheses for the true targets. Our previous approach does not perform well once incorrect measurements originating from clutter or false alarms exist, which causes multiple hypothesis of the tracked target. Recently, a hybrid approach for online joint detection and tracking for multiple targets was proposed (Ng et al., 2005). This approach does not assume the knowledge of true targets (without clutter) is given. It first uses a deterministic clustering method that searches for regions of interest (ROIs) based on the observations and monitors these ROIs for target detection, then performs multi-target tracking by Sequential Monte Carlo (SMC) methods.

In this chapter, we extend the contributed play-based tracking by introducing a multi-target dynamics model. We also take use of an improved proposal function for the PBPF based on the ROIs. We construct two multi-target trackers in the system, for the ball and the team member respectively. We use multi-target tracker instead of single target tracker because we can keep track of the true target and the false positive at the same time without losing any of them and perform ball (or team member) recognition from a pool of tracked objects later.

This chapter is organized as follows. We give a brief description of our robots and two main components of the control architecture. We describe the multi-target dynamics model. We describe the clustering algorithm we used to continuously monitor the appearance and disappearance of regions of interest (ROIs) on the field. The ROIs is used to deterministically obtain the number of targets. The ROIs is further used to get better proposal functions in particle filtering algorithm. We use an improved proposal function for the PBPF based on the ROIs. We contribute the multi-target tracking extension to the PBPF introduced in (Gu & Veloso, 2006).

Source: Multiagent Systems, Book edited by: Salman Ahmed and Mohd Noh Karsiti,
ISBN 978-3-902613-51-6, pp. 426, February 2009, I-Tech, Vienna, Austria

2. Segway RMP soccer robot

The Segway platform is unique due to its combination of wheel actuators and dynamic balancing (Browning et al., 2005). Segway RMP, or Robot Mobility Platform, provides an extensible control platform for robotics research. It imbues the robot with the novel characteristics of a fast platform and travel long ranges, able to carry significant payloads, able to navigate in relatively tight spaces for its size, and provides the opportunity to mount sensors at a height comparable to human eye level (Searock et al., 2004).
In our previous work, we have developed a Segway RMP robot base capable of playing Segway soccer. We briefly describe the two major components of the control architecture, the sensor and the robot cognition, which are highly related to our multi-model motion tracking.

2.1 Vision sensor and infrared sensors

The goal of vision is to provide as many valid estimates of targets as possible. Tracking then fuses this information to track the most interesting targets (the ball and the team member, in this paper) of relevance to the robot. We use one pan-tilt camera as the vision sensor. We do not discuss the localization of the robot in the sense that a lot of soccer tasks (known as tactics and plays described in section 2.2) can be done by the Segway RMP robot without localization knowledge. Also we use global reference for position and velocity in this paper which means it is relative to the reference point where the robot starts to do dead reckoning.
We have equipped each robot with infrared sensors to reliably detect the objects located in the catchable area of the robot. Its measurement is a binary value indicating whether or not an object is there. In most cases, this is the blind area of the pan-tilt camera. Therefore, the infrared sensor is particularly useful when the robot is grabbing the ball.

2.2 Robot cognition

A control architecture, called Skills-Tactics-Plays, was proposed in (Browning et al., 2005) to achieve the goals of responsive, adversarial team control. The key component of STP is the division between single robot behavior and team behavior.
Play, $\boldsymbol{P}$, is a fixed team plan consisting of a set of applicability conditions, termination conditions, and $\boldsymbol{N}$ roles, one for each team member. Each role defines a sequence of tactics $\boldsymbol{T_1}$, $\boldsymbol{T_2}$, … and associated parameters to be performed by that role in the ordered sequence. Assignment of roles to team members is performed dynamically at run time. Upon role assignment, each robot $\boldsymbol{i}$ is assigned its tactic $\boldsymbol{T_i}$ from the current step of the sequence for that role.
A tactic, $\boldsymbol{T}$, encapsulates a single robot behavior. Each robot $\boldsymbol{i}$ executes its own tactic as created by the current play $\boldsymbol{P}$. A tactic $\boldsymbol{T_i}$ determines the skill state machine $\boldsymbol{SSM_i}$ to be executed by the robot $\boldsymbol{i}$.
A skill, $\boldsymbol{S}$, is a focused control policy for performing some complex action. Each skill is a member of one, or more, skill state machines $\boldsymbol{SSM_1}$, $\boldsymbol{SSM_2}$, … Each skill $\boldsymbol{S}$ determines what skill it transitions to $\boldsymbol{S'}$ based upon the world state, the time skill $\boldsymbol{S}$ has been executing for, and the executing tactic for that robot.
We construct the robot cognition using a similar architecture. Plays, tactics, and skills, form a hierarchy for team control. Plays control the team behavior through tactics, while tactics

encapsulate individual robot behavior and instantiate actions through sequences of skills. Skills implement the focused control policy for actually generating useful actions. Figure 1 shows the ***SSMs*** and transitions for an example tactic: ***CatchKickToGoal***, which contains six skills. Each node in the figure is a skill and the edges show the transition between skills.

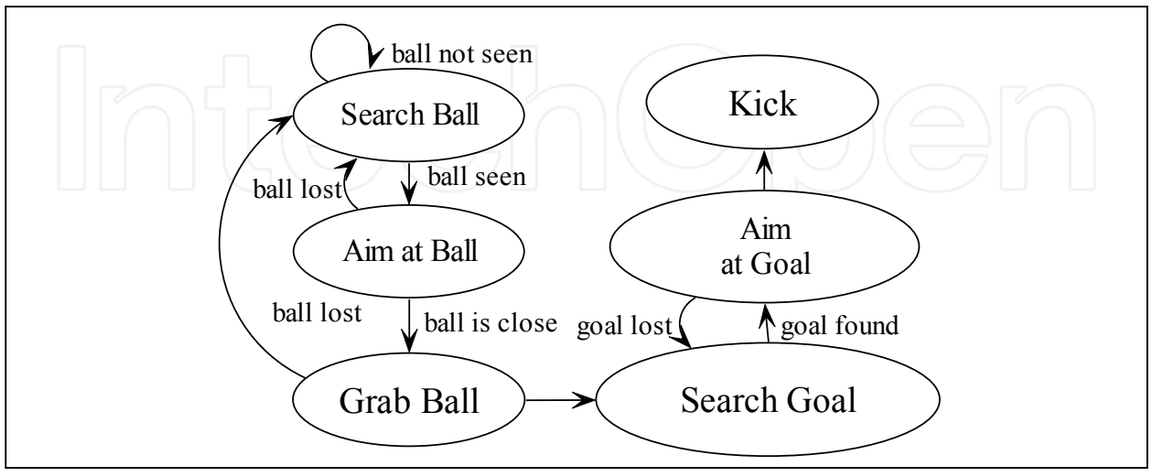

Fig. 1. Skill state machines (SSMs) for an example tactic: ***CatchKickToGoal***.

Segway soccer is a team sport, and therefore the building of our game strategy required not only execution of single robot behaviour, but also coordination with the team member, the human player. The current coordination is simple and basically based upon two fixed plays for offensive and defensive situation respectively. Our offensive play is shown as follows, in which the termination condition is either play aborted or the situation changed (a turn-over of ball possession announced by the referee). There are two roles in this play, one passes the ball to the other who positions down field and waits for receiving a pass. Once the positioning is done (e.g., the other is closer to the opponent goal), the passing is performed.

PLAY Naive Offense
APPLICABLE offense
DONE aborted OR !offense
ROLE 1
pass 2
none
ROLE 2
position_down_field
receive_pass

Table 1. The example play: ***Naive Offense***.

Our current coordination is purely observation based. Each player assigns role from his own eyeshot without communication. For example, should the robot think the team member is closer to the ball, the robot (ROLE 2) would choose to position and receive the ball from its team member (ROLE 1). Furthermore, the robot knows which side gains possession of the ball from the referee announcement, therefore it tells offensive from defensive situation clearly and thus it has deterministic idea of which play the team is using. The robot makes

an assumption that its team member is performing the same game play as itself. The robot infers what tactic the team member is executing from the team play. For example, after receiving the ball from the team member, as a passer the robot would assume the team member go forward to a tactically advantageous position to receive a pass. The predefined play for team coordination helps motion modeling, which will be further discussed in section 3.

3. Play-based motion modeling

In this section, we take a multi-target tracking problem as a detailed example to show the extension of the tactic-based motion modeling method in general when the team coordination knowledge (play) is incorporated. First we give an introduction of the environment and targets under the Segway soccer setup. Second, we describe detailed motion models for both the ball and the team member. Third, we extend the tactic-based motion modeling to the play level when both the ball and the team member are included into the tracking. We show how we model the play-dependent interactions between the team member, the robot and the ball and set up a base for giving the multi-model tracking algorithm in the next section. Although we present the Segway soccer domain as an example, the formalism is general.

3.1 Tracking scenario

Many tracking scenarios involve multiple moving targets. In a Segway soccer game, we need to track the ball, the human team member and the two opponents. Each team is identified by their distinct color. The ball is orange (Veloso et al., 2005). An observation from the sensors might consist of multiple measurements due to the moving objects and the clutter. Generally the clutter has similar color as the targets we are interested in and it causes multiple hypotheses for the true targets. Therefore, we construct two multi-target trackers in the system, for the ball and the team member respectively. We use two separate trackers instead of one because we can differentiate the ball with the team member thanks to the color-based vision system. We use multi-target tracker instead of single target tracker because we can keep track of the true target and the false positive at the same time without losing any of them and perform ball(or team member) recognition from a pool of tracked objects later.

3.2 State space and dynamics

Let $x_t = [x_{1,t}^T, x_{2,t}^T, \cdots, x_{K_t,t}^T]^T$ denote a combined target state vector for K_t unknown and time-varying number of targets. The general parameterized system process for the kth target $\mathbf{x}_{k,t}$ at time t is given by:

$$x_{k,t} = f_k\,(x_{k,t-1},\, u_{k,t-1},\, v_{k,t-1}) \tag{1}$$

Where f_k is the parameterized state transition functions for the kth target; **x, u** are the state and input vectors; **v** is the process noise vectors of known statistics. Given the number of targets at ***t-1*** and ***t***, the state transition can be represented as follows:

$$p(x_t \mid x_{t-1}, K_t, K_{t-1}) = \begin{cases} p_0(x_{K_t,t}) \prod_{k=1}^{K_{t-1}} p(x_{k,t} \mid x_{k,t-1}), if \quad K_t = K_{t-1} + 1 \\ \prod_{k=1}^{K_t} p(x_{k,t} \mid x_{k,t-1}), if \quad K_t = K_{t-1} \\ \prod_{k=1,k\neq k^*}^{K_{t-1}} p(x_{k,t} \mid x_{k,t-1}), if \quad K_t = K_{t-1} - 1 \\ 0, otherwise \end{cases} \tag{2}$$

The prior $p(x_{k,t} \mid x_{k,t-1})$ can be evaluated from Equation (1). The function $p_0(x_{Kt,t})$ is the initial distribution of a new target state and k^* is a target that vanishes at t. We assume at most one target appears or vanishes in each time step. The number of targets is obtained deterministically (see section 4.1 for the details).

3.3 Observation model

We assume that measurements are independent of each other and either originate from true targets or clutter. Furthermore, each target generates at most one measurement in each time step but may not be detected by the sensor.

Let $z_{c,t} = [z_{1,t}^T, z_{2,t}^T, \cdots, z_{M_t,t}^T]^T$ denote an observation vector for M_t measurements. The observation equation for modeling the cth measurement originating from the kth target is given as:

$$Z_{c,t} = h_k (x_{k,t}, n_{c,t}) \tag{3}$$

where $\mathbf{n}_{c,t}$ is a zero-mean observation noise with a known covariance Σ_n. The measurement originating from a clutter is modelled to be uniformly distributed within the entire visible region (observation volume V) of the camera. We use a deterministic method to perform measurement-to-target association. Let α_t denote the association vector which indicate the measurement-to-target assignment, whose ith element $\alpha_{i,t}$ is set to j if the ith measurement originates from the jth target, or zero if it originates from clutter. Let N_{Ct} denote the number of clutter points, and Ω_D denote the set of measurements indices corresponding to the detected targets, the likelihood function for measurement $\mathbf{z}_t$ can be written as:

$$p(z_t \mid x_t, K_t, \alpha_t) = \left(\frac{1}{v}\right)^{N_{C_t}} \prod_{l \in \Omega_D} p(z_{l,t} \mid x_{\alpha_{l,t},t}), \tag{4}$$

where $p(z_{l,t} \mid x_{\alpha_{l,t},t})$ is evaluated from (3).

3.4 Ball motion modeling

In our Segway RMP soccer robot environment, we define five models to model the ball motion (for the rest of this paper, for simplicity, we use $\mathbf{x}_t$ to represent the ball state, and use $\mathbf{x}'_t$ to represent the team member state).

- ***Free-Ball***, ***Robot-Grab-Ball***, ***Robot-Kick-Ball***. We use the same models as ***Free-Ball***, ***Grab-Ball***, ***Kick-Ball*** introduced in (Gu, 2005) respectively.
- ***Human-Grab-Ball.*** The ball is held by the team member. We can infer the ball position similarly if we know the team member position well.
- ***Human-Grab-Ball.*** The ball is kicked by the team member and it is supposed to be either a pass to the robot or a shoot at the goal.

In general, we will have n motion models $m_1, m_2, \ldots, m_n$.

3.5 Team member motion modeling

We define four models to model the human team member's motion.

- ***Random Walk***. The team member is wandering in the field. So the state at the new time is the state at the current time with some additive zero-mean (assumed Gaussian) noise.
- ***Holding Ball***. The team member is holding the ball without moving and waiting for the robot to receive the ball. Should the robot know the ball position well, it can infer the team member position by the ball position in a similar way as ***Robot-Grab-Ball*** for ball motion modeling.
- ***Accelerating (Decelerating)***. The team member dashes (stops) to obtain (lose) a velocity in a short time.
- ***Positioning***. The team member is going to a predefined tactical position with a constant speed. This case happens mostly after the team member passing the ball to the robot and moving down the field toward opponent's goal.

3.6 Play based model transitions

Given the knowledge of the team coordination plan (the play P_{t-1} at time $t-1$), the robot can infer what tactic the team member is executing (T'_{t-1}), which provides valuable information about the motion model of the team member (m'_t). Both the robot and the team member act over the ball in a Segway soccer game. The motion model of the ball (m_t) is therefore affected by what tactic the robot (T_{t-1}) and the team member (T'_{t-1}) are executing.

From the previous subsection, we know that the model index m determines the present model being used. For our team member tracking example, $m'_t = i$, $i = 1, \ldots, 4$\$. In our approach, it is assumed that the team member motion model index, m'_t, conditioned on the previous tactic executed T'_{t-1} by the team member, and other useful information v'_t (such as ball state), is governed by an underlying Markov process, such that, the conditioning parameter can branch at the next time-step with probability.

$$p(m'_t = i \mid m'_{t-1} = j, T'_{t-1}, v'_t) = h'_{i,j}, \tag{5}$$

where $i, j = 1, \ldots, N_{m'}$. Since T'_{t-1} can be determined by P'_{t-1}, we get

$$h'_{i,j} = p(m'_t = i \mid m'_{t-1} = j, P_{t-1}, v'_t). \tag{6}$$

Since we can draw $p(m'_t = i \mid m'_{t-1} = j)$ in an $N_{m'} \times N_{m'}$ table, we can create a table for Equation (6) with a third axis which is defined by the $\langle P_a, v_b \rangle$ as shown in Figure 2. Here the play P_a, is the primary factor that determines whether m_i transits to m_j and what the transition probability is, while the information V_b determines the prior condition of the transition. Each layer in the graph is conditioned on a particular combination of the tactic executed and the additional information obtained.

For our ball tracking example, $m_t = i$, $i = 1, \ldots, 5$. Similarly,

$$h_{i,j} = p(m_t = i \mid m_{t-1} = j, T_{t-1}, T'_{t-1}, v_t), \tag{7}$$

where $i, j = 1, \ldots, N_m$. Since T_{t-1}, T'_{t-1} can be determined by P_{t-1}, we get

$$h_{i,j} = p(m_t = i \mid m_{t-1} = j, P_{t-1}, v_t) \tag{8}$$

Suppose the current team play is the ***Naive Offense*** in Section 2.2, we can obtain the corresponding motion model transitions for both the ball and the team member using the play-based method (Figure 3).

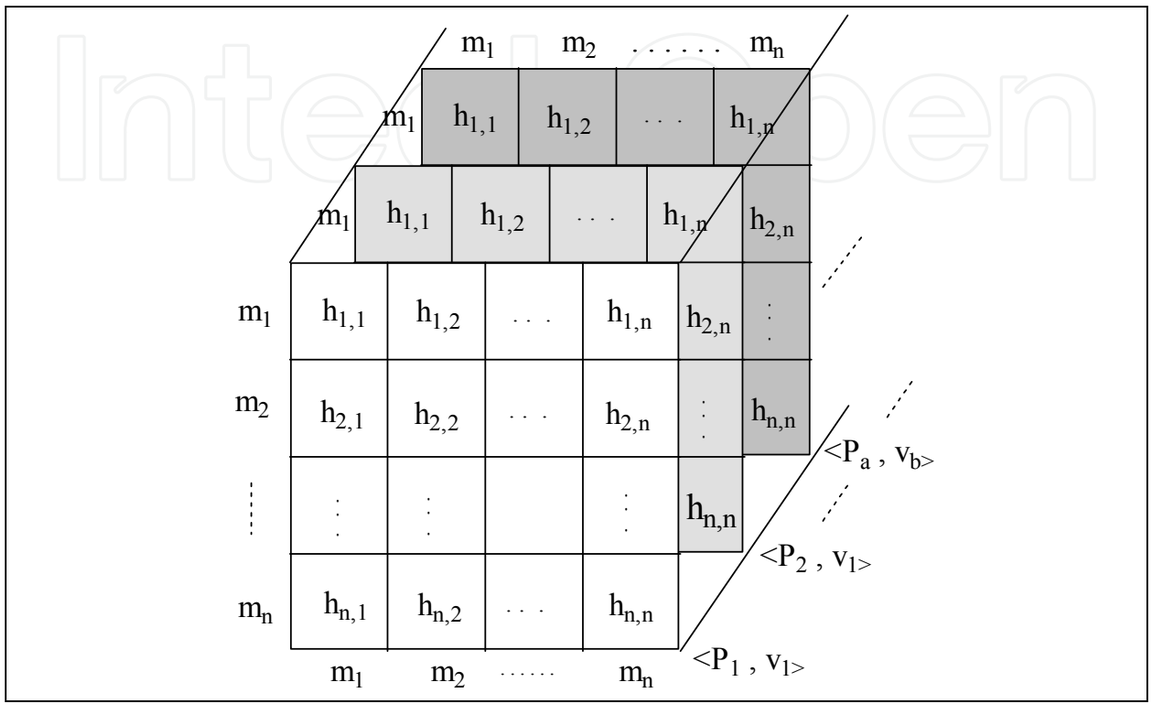

Fig. 2. Play-Based motion modeling. Each layer in the graph is conditioned on a particular combination of the play executed and the additional information obtained.

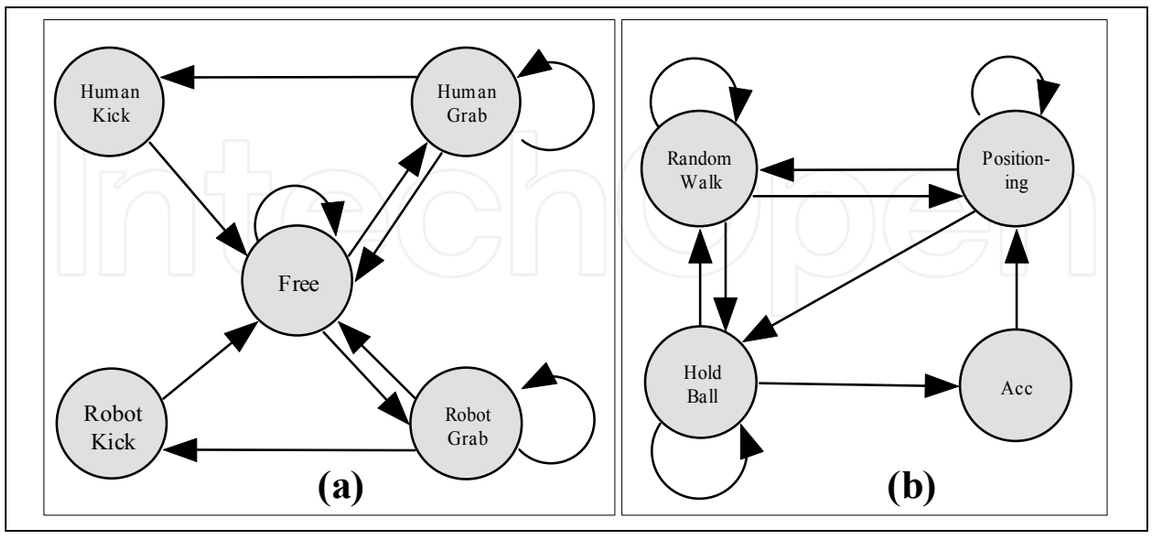

Fig. 3. Object motion modelling based on the play: Naive Offense. Each node is a model. Models transit to one another according to the predefined probabilities (not shown in the figures). (a) Ball motion model. (b) Human team member motion model.

4. Multi-model motion tracking

In this section, we first describe the clustering algorithm we used to continuously monitor the appearance and disappearance of regions of interest (ROIs) on the field. We then use dynamic Bayesian networks to represent the whole system. We give the detailed sequential Monte Carlo methods of tracking in a multi-model multi-hypothesis scheme finally.

4.1 Clustering algorithm

The idea of this algorithm is to group a set of ROIs, $S_t = \{S_t^{(j)}\}_{j=1}^{J_t}$, within a buffer of observations, $Z_t = \{z_{t'}\}_{t-\tau}^{t}$, where τ is the length of the sliding window (Ng et al., 2005). If the current targets are widely separated, the measurements originating from them will be clustered in the locations where the targets had visited from $t-\tau$ to t. Therefore the distance between two measurements within two successive time steps can be taken as the clustering criteria. Let $d_{c,l}(t'+1,t')$ denote the normalized distance between the *cth* and *lth* measurements of $z_{t'+1}$ and $z_{t'}$.

$$d_{c,l}(t'+1,t') = e_{c,l}^{T}(t'+1,t') \sum\nolimits_{e}^{1} e_{c,l}(t'+1,t')\, t \tag{9}$$

where $c \in \{1,\cdots,M_{t'+1}\}$ and $l \in \{1,\cdots,M_{t'}\}$, $e_{c,l}(t'+1,t') = z_{c,t'+1} - z_{l,t'}$ and $\sum_e = 2\sum_n$. For all measurements of $z_{t'+1}$, we have a set of normalized distances $\{d_{c,l}(t'+1,t')\}_{c=1}^{M_{t'+1}}$. The *c*th* measurement $z_{c^*,\,t'+1}$ will be grouped with $z_{l,\,t'}$ in the same cluster $S_t^{(j)}$ if

$$d_{c^*,l}(t'+1,t') = \arg\min\{d_{c,l}(t'+1,t')\}_{c=1}^{M_{t'+1}} \tag{10}$$

and

$$d_{c^*,l}(t'+1,t') \le \varepsilon_z, \tag{11}$$

where ε_z is a threshold. If none of the measurements of $z_{t'+1}$ can be grouped with $z_{l,\,t'}$, the search procedure continues and uses $z_{t'+b}$, $1 < b \le \tau$ until (10) and (11) are both satisfied. Meanwhile the algorithm propagates through time to group other measurements in the remaining observations. Since the detected regions obtained via (10) and (11) possibly arise from clutter points, another threshold is set to examine the number of measurements in the region to eliminate the false positives. That is, if the number of measurements in the *jth* region $S_t^{(j)}$ is less than τ_{min}, defined as the minimum number of clustered measurements in a region required to identify a target, it is discarded; otherwise, the *jth* region is classified as originating from a target. Once a set of ROIs $S_t = \{S_t^{(j)}\}_{j=1}^{J_t}$ is obtained, it is necessary to determine which region belongs to which existing active track or a new track deterministically. Let $\gamma_t = [\gamma_{1,t}, \gamma_{2,t}, \cdots, \gamma_{K_t,t}]^T$ denote the track-to-region association vector. $\gamma_{k,t}$ is used to indicate the association between the ROIs and the active tracks. $\gamma_{K_t,t}$ is evaluated as *j* if track *k* can be associated with $S_t^{(j)}$, otherwise it is zero. Refer to (Ng et al., 2005) for more details of the clustering algorithm.

4.2 DBN representation

Following the play-based motion model, we can use dynamic Bayesian networks (DBNs) to represent the whole system for team member and ball tracking in a natural and compact way as shown in Figure 4 and Figure 5 respectively. In the two graphs, the system state is represented by variables (play $\boldsymbol{P}$, tactic $\boldsymbol{T}$, infrared sensor measurement $\boldsymbol{s}$, ball state $\mathbf{x}$, ball motion model index $\boldsymbol{m}$, vision sensor measurement of ball $\mathbf{z}$, team member state $\mathbf{x'}$, team member motion model index $\boldsymbol{m'}$, vision sensor measurement of team member $\mathbf{z}$), where each variable takes on values in some space. The variables change over time in discrete intervals, so that e.g., $\mathbf{x}_t$ is the ball state at time $\boldsymbol{t}$.

Furthermore, the edges indicate dependencies between the variables. For instance, in Figure 5 the ball motion model index $\boldsymbol{m}_t$ depends on $\boldsymbol{m}_{t-1}$, $\boldsymbol{T}_{t-1}$, $\boldsymbol{T'}_{t-1}$, $\boldsymbol{s}_t$ and $\mathbf{x}_{t-1}$, hence there are edges coming from the latter five variables to $\boldsymbol{m}_t$. For the rest of this section, we give the ball-tracking algorithm following Figure 5. The team-member-tracking algorithm can be obtained similarly following Figure 4.

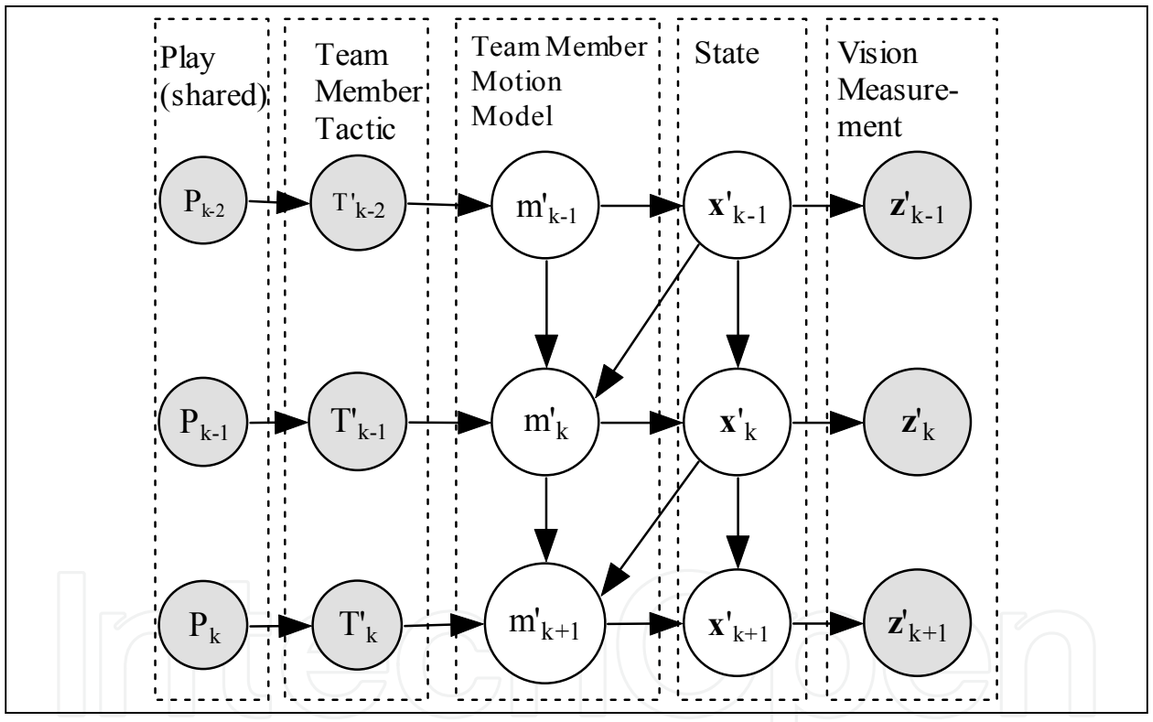

Fig. 4. A dynamic Bayesian network for team member tracking. Filled circles represent deterministic variables with are observable or are known as tactics or plays that the robot is executing.

4.3 Importance sampling function

We use the sequential Monte Carlo method to track the motion model $\boldsymbol{m}$ and the multi-target state $\mathbf{x}$. Particle filtering is a general purpose Monte Carlo scheme for tracking in a dynamic system (Doucet et al., 2001). It maintains the belief state at time $\boldsymbol{t}$ as a set of particles $p_t^{(1)}, p_t^{(2)}, \cdots, p_t^{(N_s)}$, where each $p_t^{(i)}$ is a full instantiation of the tracked variables

$\{p_t^{(i)}, w_t^{(i)}\}$, $w_t^{(i)}$ is the weight of particle $p_t^{(i)}$ and N_s is the number of particles. In our case, $p_t^{(i)} = \left\langle x_t^{(i)}, m_t^{(i)} \right\rangle$. To make our notation more concrete, a particular particle $p_t^{(i)}$, which is tracking Kt multi-target state vector xt and motion model mt, is given as (Kreucher et al., 2003):

$$p_t^{(i)} = \begin{pmatrix} m_{1,t}^{(i)} & m_{2,t}^{(i)} & \cdots & m_{K_t,t}^{(i)} \\ x_{1,t}^{(i)} & x_{2,t}^{(i)} & \cdots & x_{K_t,t}^{(i)} \\ y_{1,t}^{(i)} & y_{2,t}^{(i)} & \cdots & y_{K_t,t}^{(i)} \\ \dot{x}_{1,t}^{(i)} & \dot{x}_{2,t}^{(i)} & \cdots & \dot{x}_{K_t,t}^{(i)} \\ \dot{y}_{1,t}^{(i)} & y_{2,t}^{(i)} & \cdots & y_{K_t,t}^{(i)} \end{pmatrix} \tag{12}$$

We sample the ball motion model following the ball-tracking DBN as below:

$$m_{k,t}^{(i)} \sim p(m_{k,t} \mid m_{k,t-1}^{(i)}, x_{k,t-1}^{(i)}, s_t, T_{t-1}, T'_{t-1}) \tag{13}$$

Note that T_{t-1} and T'_{t-1} are inferred deterministically from P_{t-1} instead of sampling. Conditioned on the ball motion model $m_{k,t}^{(i)}$, we then use the importance function introduced in (Ng et al., 2005) to sample ball state $x_{k,t}^{(i)}$:

$$x_{k,t}^{(i)} \sim \begin{cases} q_D(x_{k,t} \mid m_{k,t-1}^{(i)}), k \notin \Psi_{S_t} \\ q_{DS}(x_{k,t} \mid m_{k,t}^{(i)}, x_{k,t-1}^{(i)}, S_t^{(j)}, k \in \Psi_{S_t} \end{cases}, \tag{14}$$

where $k \in \Psi_{s_t}$ are those tracks with $\gamma_{k,t} = j$ and $j \in \Omega_D$, and $q_D(\cdot)$ and $q_{DS}(\cdot)$ are the proposal functions for $x_{k,t}$ without and with an associated ROI $S_t^{(j)}$, given as follows, respectively,

$$q_D(x_{k,t} \mid m_{k,t}^{(i)}, x_{k,t-1}^{(i)}) = p(x_{k,t} \mid m_{k,t}^{(i)}, x_{k,t-1}^{(i)}), \tag{15}$$

$$q_{DS}(x_{k,t} \mid m_{k,t}^{(i)}, x_{k,t-1}^{(i)}, S_t^{(j)}) = \mu p(x_{k,t} \mid m_{k,t}^{(i)}, x_{k,t-1}^{(i)}) + (1-\mu) q(x_{k,t} \mid m_{k,t}^{(i)}, x_{k,t-1}^{(i)}, S_t^{(j)}), \tag{16}$$

where $0 \le \mu \le 1$ and $q(\cdot)$ is a uniform sampling from the associated ROI $S_t^{(j)}$. If $\mu = 1$, the importance sampling function is reduced back to the dynamic prior. If $\mu = 0$, all particles are generated from the data-dependent importance function. If $0 < \mu < 1$, this proposal combines the dynamic prior and the current ROIs to generate representative particles.

4.4 Birth, death and update moves

Assuming that there are K_t^b ROIs that cannot be associated with any existing track, we will initiate a new track in each time step from one of these regions instead of initiating K_t^b tracks simultaneously in order to fit the birth move with the assumed system process model in (2). When an existing track cannot be associated with a region at a given time, the target being tracked by the tracker may have disappeared or temporarily experience a short period

of data loss. Thus we may remove the track for the target only if it has failed to associate with any ROI with τ_d time steps. Refer to (Ng et al., 2005) for the detailed algorithm of birth move and death move.

In the update move, there is no change in terms of the number of ROIs. We only need to update the target states with a common value of number of targets $K_t^{(i)} = K_{t-1}^{(i)}$, using the sequential importance sampling method as follows:

$[\{x_t^{(i)}, m_t^{(i)}, w_t^{(i)}\}_{i=1}^{N_s}] = \text{MT-PBPF}[\{x_{t-1}^{(i)}, m_{t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N_s}, z_t, s_t, T_{t-1}, T'_{t-1}]$
for i = 1: N_s
  for k = 1: K_t
    draw $m_{k,t}^{(i)} \sim p(m_{k,t} | m_{k,t-1}^{(i)}, x_{k,t-1}^{(i)}, s_t, T_{t-1}, T'_{t-1})$
    if track k has corresponding ROI $S_t^{(j)}$
      draw $x_{k,t}^{(i)} \sim q_{DS}(x_{k,t} | m_{k,t}^{(i)}, x_{k,t-1}^{(i)}, S_t^{(j)})$
    else
      draw $x_{k,t}^{(i)} \sim q_D(x_{k,t} | m_{k,t}^{(i)}, x_{k,t-1}^{(i)})$
    end if
  end for
  set $w_t^{(i)} = w_{t-1}^{(i)} \quad p(z_t | x_t, K_t, \alpha_t)$
end for
calculate total weight: $w = \sum_{i=1}^{N_s} w_t^{(i)}$
for i = 1: N_s
  normalize: $w_t^{(i)} = w_t^{(i)} / w$
end for
resample

Table 2. The Multi-Target Play-Based Particle Filtering algorithm (MT-PBPF).

The inputs of the algorithm are samples drawn from the previous posterior $\langle x_{t-1}^{(i)}, m_{t-1}^{(i)}, w_{t-1}^{(i)} \rangle$, the present vision and infrared sensory measurement z_t, s_t,the robot's tactic T_{t-1}, and the team member's tactic T'_{t-1}. The outputs are the updated weighted samples $\langle x_t^{(i)}, m_t^{(i)}, w_t^{(i)} \rangle$. In the sampling algorithm, first, a new ball motion model index, $m_t^{(i)}$, is sampled according to (13) at line 03. Then given the model index, and previous ball state, a new ball state is sampled according to (14) at line 05/07. According to (4), the importance weight of each sample is given by the likelihood of the vision measurement given the predicted new ball state at line 10. Finally, each weight is normalized and the samples are resampled. Then we can estimate the ball state based on the mean of all the $x_t^{(i)}$. Though we are trying to eliminate the clutter from the beginning of tracking (clustering algorithm), due to the property of the multi-target tracker, further recognition process might be done in order to figure out which tracked target is the true ball. Similarly the state of the team member $\mathbf{x}'_t$ can be obtained from the team member tracker.

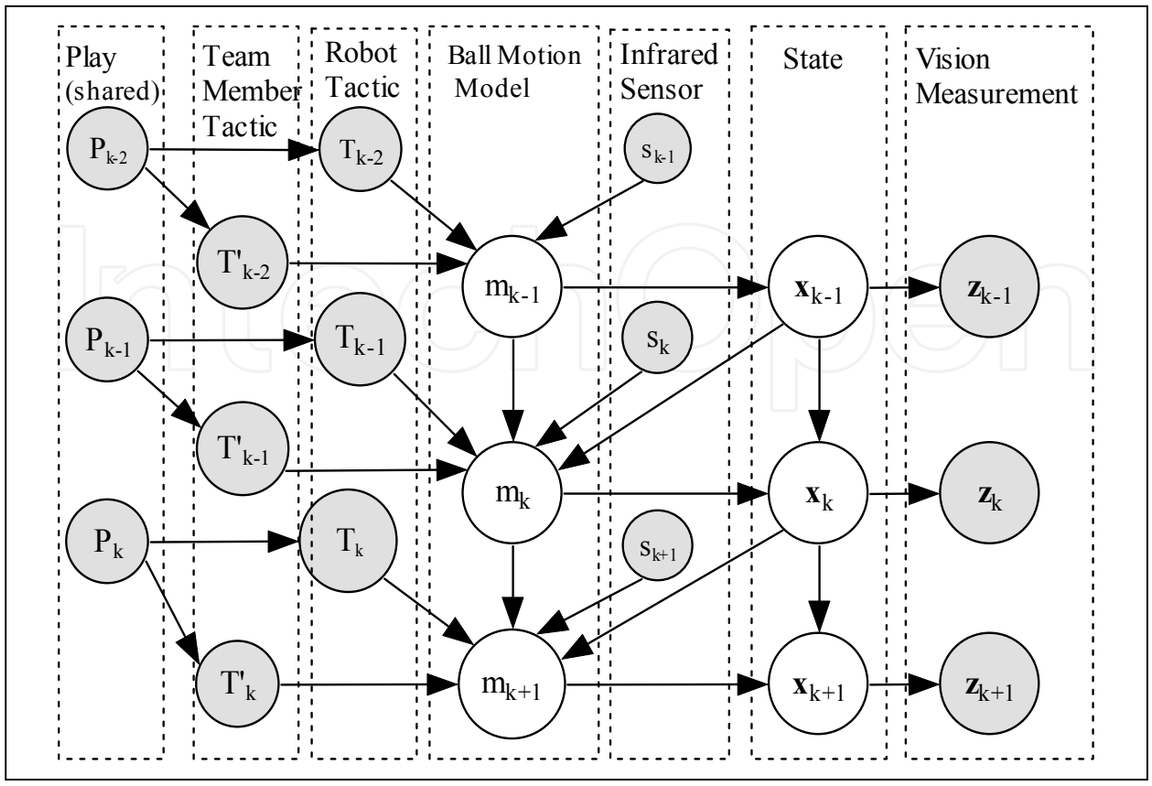

Fig. 5. A dynamic Bayesian network for object tracking.

5. Experimental results

From previous work we knew the initial speed and accuracy of the ball velocity after a kick motion. We profiled the system and measurement noise as well. In this section, we evaluate the effectiveness of our tracking system in both simulated and real-world tests.

5.1 Simulation experiments

Because it is difficult to know the ground truth of the target's position and velocity in the real robot test, we do the simulation experiments to evaluate the precision of tracking.

Motion Model	Single Model	Multi-Model
Human Position Est RMS (m)	0.0030	0.0014
Human Velociy Est RMS (m/s)	0.42	0.025
Ball Position Est RMS (m)	0.0028	0.0017
Ball Velocity Est RMS (m/s)	0.4218	0.0597

Table 3. The average RMS error of position estimation and velocity estimation from human trackers and ball trackers.

Experiments are done following the ***Naive Offense*** play, in which the robot acts as the receiver and the human team member acts as the passer. Noises are simulated according to the model we profiled in previous work. In the beginning, the team member holds the ball. After a fixed amount of time, the ball is kicked towards the robot, and the team member moves forward to a predefined location.

We implement both a single model tracker and a play-based multi-model tracker for the ball and the team member. We simulate the experiment for 50 runs, and then compare the performance of the two trackers with different implementations. The average RMS error of position estimation and velocity estimation are shown in Table 3. The results show that the play-based multi-model scheme performs much better than the single model especially in terms of velocity estimation. Because with the play-based motion model, when the ball is being kicked, most particles evolving using the transition model determined by the play will change its motion model $m_t^{(i)}$ from ***Free-Ball*** to ***Human-Kick-Ball***, and a velocity will be added to the ball accordingly.

5.2 Multi-target tracking test

In this test, one Segway RMP robot is tracking one or more balls on the field with ***SearchBall*** tactic. We would like to compare solely the target detection performance between the proposed method and the IMM tracker. A scenario with K_t ($0 \le K_t \le 3$) balls appearing and disappearing at different times and there are a set of false positives at fixed position in the surroundings.

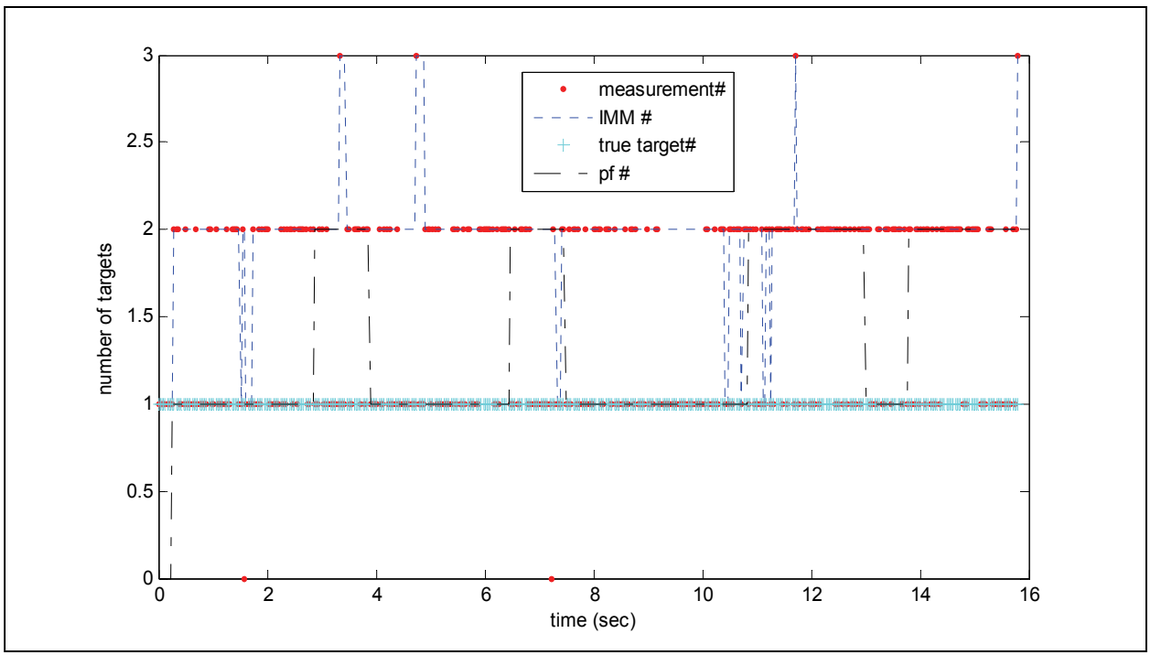

Fig. 6. A comparison of the target detection performance between the proposed method and the IMM tracker when only one target exists with surrounding clutters.

When estimating the number of targets, 3600 particles are used in the proposed method. Figures 6-7 summarize the results. In both figures, the dots show the number of

measurements at a given time. The dotted line represents the number of the targets tracked by the IMM tracker. The dash dotted line represents the number of targets tracked by the multi-model multi-target tracker proposed in this paper. The crosses show the true number of the targets at any given time. As shown in the figure, the IMM tracker is sensitive to the number of measurements, while our approach is more robust and consistent to high clutter density. Since the detection is basically performed on the clustering of the observations and the association between the detected ROIs and the existing tracks, it is computational low-cost. Therefore it is also practical for real-time multi-target detection and tracking.

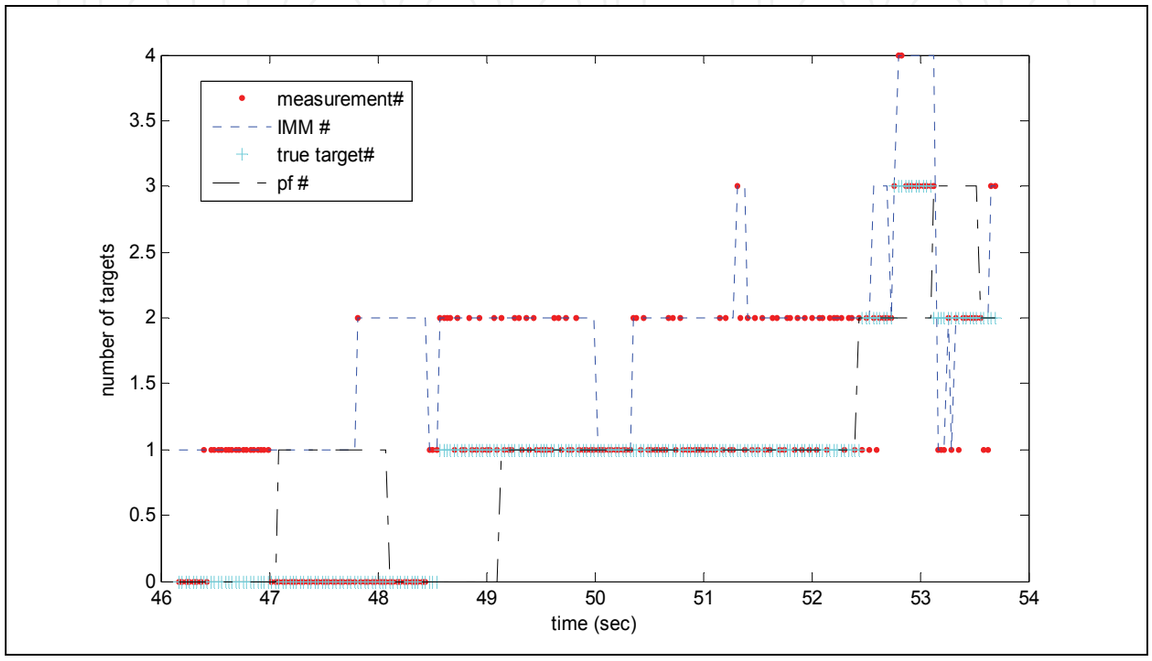

Fig. 7. A comparison of the target detection performance between the proposed method and the IMM tracker when multiple targets exist with surrounding clutters.

5.3 Team cooperation test

We do experiments on the Segway RMP soccer robot executing the offensive play and coordinating with the human team member. The test setup is demonstrated in Figure 8, in which the digits along the lines show the sequence of the whole strategy, the filled circle at position B represents the robot, the unfilled circle at position E represent an opponent player, and the shaded circle represent the human team member.

When each run begins, the human team member is at position A. With this team cooperation plan (play), the robot chooses the tactic ***CatchKickToTeammember*** to execute, in which the robot starts with the skill ***Search-Ball***. When the robot finds the ball, the team member passes the ball directly to the robot and chooses a positioning point to go to either at C or D. The robot grabs the ball after the ball is in the catchable area and is detected by the infrared sensor (skill ***Grab-Ball***). Next the robot searches for the team member holding the ball with its catcher (skill ***Search-Teammember***). After the robot finds the team member, the robot kicks the ball to its team member (skill ***KickToTeammember***) and the team member shoots at the goal, completing the whole offensive play. Each run ends in one of the following conditions.

- Succeed if the human receives the ball from the robot or the human does not receiver the ball but the pass can be considered as a "good" one.
- Fail if the robot is in searching for the ball or the team member for more than 30 seconds.
- Fail if the ball is outside the field before the robot catches it.

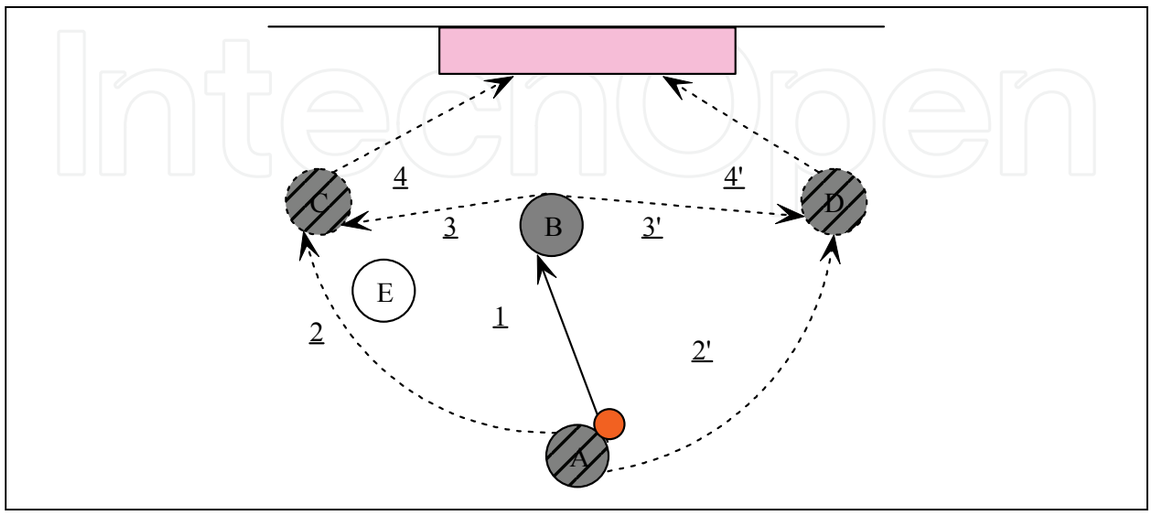

Fig. 7. A comparison of the target detection performance between the proposed method and the IMM tracker when multiple targets exist with surrounding clutters.

In the experiment over 15 runs, the robot with single model trackers fails 5 of the total. While the robot with play-based multi-model trackers fails 2 of the total. We also keep track of the mean time taken in all the successful runs. We list the result in Table 4. Using play-based multi-model tracking saves 32.3% time in terms of completing the whole play over single model tracking. During the experiment, we note that when using the single model tracking, most time was spent on searching the team member. Incorporating the team cooperation knowledge known as play into the team member motion modeling greatly improves the accuracy of the team member motion model and therefore avoids taking time in searching a lost target from scratch.

Motion Model	Single Model	Multi-Model
Mean Time (sec)	33.4	22.6

Table 4. The average time taken over all the successful runs.

6. Related work

Tracking moving targets using a Kalman filter is the optional solution if the system follows a single model, $\boldsymbol{f}$ and $\boldsymbol{h}$ in Equation (1) and (3) are known linear functions and the noise $\boldsymbol{v}$ and $\boldsymbol{n}$ are Gaussians (Arulampalam et al., 2002). Multiple model Kalman filters such as Interacting Multiple Model (IMM) are known to be superior to the single Kalman filter when the tracked target is manoeuvring (Bar-Shalom et al., 2001). For nonlinear

systems or systems with non-Gaussian noises, a further approximation is introduced, but the posterior densities are therefore only locally accurate and do not reflect the actual system densities.

Since the particle filter is not restricted to Gaussian densities, a tactic-based motion modeling method is proposed in (Gu, 2005). Based on that approach, we further introduce the play-based motion modeling method when team coordination knowledge is available. Another related approach was proposed to track a moving target using Rao-Blackwellised particle filter (Kwok & Fox, 2004) in which a fixed transition table was used between different models. Our transition model is dependent on the play that the robot is executing and the additional information that matters. This model can be flexibly integrated into our existing STP architecture.

There have been different strategies in multi-target tracking. In order to handle the data association and tracking problem, the classical Joint Probabilistic Data Association Filter (JPDAF) adopts the methods like the extended Kalman Filter (EKF) for multi-target state estimation, whose tracking performance is known to be limited by the linearity of the data models (Bar-Shalom & Fortmann, 1988). Another approach known as sequential Monte Carlo methods is able to perform well even when the data models are nonlinear and non-Gaussian. However, almost all of these methods assume that the knowledge of true targets (without clutter) is given, which is not applicable in the field that Segway RMP soccer robots operates in.

Recently, a hybrid approach for online joint detection and tracking for multiple targets was proposed (Ng et al., 2005). This approach does not rely on the clutter-free assumption. In this paper, based on their approach, we present a play-based multi-target tracking algorithm, which incorporates tactic information to eliminate the false alarms and to improve resampling efficiency. Compared to our method, first, existing techniques consider less complex dynamic systems where only one part of the state space is non-linear. In contrast, our approach estimates a system where multiple components are highly non-linear (Segway RMP robot motion, ball motion, team member motion). Second, most existing techniques examine their performance with simulated experiments, while we test our approach in real robot experiments. Third, our approach goes beyond existing techniques by incorporating team cooperation information into the tracking process which further improves the performance.

7. Conclusions and future work

Motivated by the interactions between a team and the tracked target, we contribute a method to achieve efficient tracking through using a play-based motion model and combined vision and infrared sensory information. This method gives the robot a more exact task-specific motion model when executing different tactics over the tracked target (e.g. the ball) or collaborating with the tracked target (e.g. the team member). Then we represent the system in a compact dynamic Bayesian network and use particle filter to keep track of the motion model and target state through sampling. The empirical results from the simulated and using the real robot agent show the efficiency of the multi-model tracking over single model tracking.

If the teammate is a human, not a robot, the certainty that the teammate is executing the expected play or tactic could be reduced. That is, the human teammate could fail to execute the desired play or tactic. Future work will take such uncertainty into account. A better human team member modeling (for example, include intercepting the moving ball, mark a player, covering the goal) will also help. Another interesting work is to know how the performance of the presented method is affected by the presence of tactics of the team member that are not exactly determined in the team coordination plan.

8. Acknowledgments

We would like to thank the members of the CMBalance Segway soccer team for their help with developing the infrastructure for the Segway robots. This work was supported by United States Department of the Interior under Grant No. NBCH-1040007. The content of the information in this publication does not necessarily reflect the position or policy of the Defense Advanced Research Projects Agency (DARPA), US Department of Interior, US Government, and no official endorsement should be inferred.

9. References

S. Arulampalam; S. Maskell, N. Gordon, T. Clapp (2002). A tutorial on particle filters for on-line non-linear/non-gaussian Bayesian tracking. *IEEE Transactions on Signal Processing,* 50(2):174–188, Feb.2002.

Y. Bar-Shalom & T. E. Fortmann (1988). *Tracking and Data Association.* Academic Press, Inc, 1988.

Y. Bar-Shalom; X.-R. Li, & T. Kirubarajan (2001). *Estimation with Applications to Tracking and Navigation.* John Wiley & Sons, Inc, 2001.

B. Browning; J. Bruce; M. Bowling & M. Veloso (2005). STP: Skills, tactics and plays for multi-robot control in adversarial environments. *IEEE Journal of Control and Systems Engineering,* 219:33–52, 2005.

B. Browning; J. Searock; P. E. Rybski & M. Veloso (2005). Turning segways into soccer robots. *Industrial Robot,* 32(2):149–156, 2005.

A. Doucet; N. D. Freitas & N. Gordon (2001). *Sequential Monte Carlo Methods in Practice.* Springer-Verlag, New York, 2001.

Y. Gu (2005). Tactic-based motion modelling and multi-sensor tracking. *Proceedings of Twentieth National Conference on Artificial Intelligence,* 2005.

C. Kreucher; K. Kastella & A. O. H. III (2003). Multi-target sensor management using alpha-divergence measures. pp 209–222, 2003.

C. Kwok & D. Fox (2004). Map-based multiple model tracking of a moving object. *Proceedings of eight RoboCup International Symposium,* July 2004.

W. Ng; J. Li, S. Godsill, & J (2005). Vermaak. A hybrid approach for online joint detection and tracking for multiple targets. *IEEE Aerospace Conferences,* 2005.

D. Schulz; W. Burgrad & D. Fox (2003). People tracking with mobile robots using sample-based joint probabilistic data association filters. *International Journal of Robotics Research,* 22(2), 2003.

J. Searock; B. Browning & M. Veloso (2004). Turning Segways into Soccer Robots. *In Proceedings of IROS'04,* Sendai, Japan, September 2004.

M. Veloso; B. Browning; P. Rybski & J. Searock (2005). Segwayrmp robot football league rules. Technical report, http://www.cs.cmu.edu/ robosoccer/segway/, 2005.

Multiagent Systems

Edited by Salman Ahmed and Mohd Noh Karsiti

ISBN 978-3-902613-51-6
Hard cover, 426 pages
Publisher I-Tech Education and Publishing
Published online 01, January, 2009
Published in print edition January, 2009

Multi agent systems involve a team of agents working together socially to accomplish a task. An agent can be social in many ways. One is when an agent helps others in solving complex problems. The field of multi agent systems investigates the process underlying distributed problem solving and designs some protocols and mechanisms involved in this process. This book presents an overview of some of the research issues in the field of multi agents. It is a presentation of a combination of different research issues which are pursued by researchers in the domain of multi agent systems as they are one of the best ways to understand and model human societies and behaviours. In fact, such systems are the systems of the future.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen