Locating Moving Entities in Indoor Environments with Teams of Mobile Robots

Matthew Rosencrantz
mrosen@cs.cmu.edu

Geoffrey Gordon
ggordon@cs.cmu.edu

Sebastian Thrun
thrun@cs.cmu.edu

Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213

ABSTRACT

This article presents an implemented multi-robot system for playing the popular game of laser tag. The object of the game is to search for and tag opponents that can move freely about the environment. The main contribution of this paper is a new particle filter algorithm for tracking the location of many opponents in the presence of pervasive occlusion. We achieve efficient tracking principally through a clever factorization of our posterior into roles that can be dynamically added and merged. When searching for opponents, the individual agents greedily maximize their information gain, using a negotiation technique for coordinating their search efforts. Experimental results are provided, obtained with a physical robot system in large-scale indoor environments and through simulation.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Probabilistic Algorithms; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent systems*

General Terms

Algorithms

Keywords

Particle Filter, Robot Exploration, Laser tag

1. INTRODUCTION

This paper describes a multi-robot system capable of locating and pursuing moving objects (people, other robots) in indoor environments. Our research is motivated by the popular game "laser tag" [27]. The object of laser tag is to find and tag individuals from an opposing team using an infra-red tagging device. The robotic form of lasertag involves teams of robots instead of people. In our implementation, it is played in regular buildings, with Pioneer-size robots equipped with laser range sensors (see Figure 1).

Figure 1: The laser tag robots are equipped with light-emitting devices and receivers.

The game of laser tag provides unique opportunities for multi-agent robotics research. Just like robotic soccer [12], laser tag is inherently real-time. The environment is dynamic, leaving only limited time for information fusion and decision making. A key difference between lasertag and robotic soccer arises from the pervasive presence of occlusion: most of the time neither teammates nor opponents are within a robot's sensor field. To play well, the robots must carefully keep track of where possible opponents might be, and which areas of the environment have already been observed. This creates a challenging multi-robot data fusion problem. Furthermore, the robots have to coordinate their actions in the face of this uncertainty.

As mentioned above one of the main problems in lasertag is tracking multiple moving objects under pervasive occlusion. This problem, however, has received relatively little attention in the literature. This paper seeks to address this question. The core of this paper is an algorithm for maintaining posteriors over the locations of moving objects in environments where occlusion is pervasive. In our approach we use particle filters [7, 14] because they allow the team to represent complex multi-modal posteriors.

Our technique is based on two related insights. First, pervasive occlusion leads directly to an intractable data association problem. Data association is often difficult, but in many cases if the objects being tracked are well separated and frequently observed, approximating data association with a maximum likelihood point estimate is effective. In lasertag and other domains where occlusion is pervasive, this approximation is poor because many associations are plausible. The second insight is that the same pervasive occlusion that makes data association hard introduces structure into our posterior. Our sensors' inability to see through walls and around corners causes many dimensions of our belief to become practically equivalent. We show how to capitalize on this redundancy by categorizing our opponents into *roles*. Roles allow us to represent groups of opponents by their known characteristics, ignoring their individual identities. This role based factorization leads to a tracker that can grow and shrink, adding or removing dimensions from its state space as new roles are formed or old ones become indistinguishable. At the same time, focusing on observable characteristics drastically simplifies our data association problem. We show empirically that there are situations where the number of opponents is large, but the number of active roles remains quite small.

In the following sections we will develop an efficient particle filter tracking algorithm by using the role-based factorization described above. We will proceed to demonstrate mathematically that our factorization holds even as more observations are received and controls issued. We will also present a greedy algorithm for issuing coordinated controls to our teammates, and use it to demonstrate that our tracker is practically useful in the laser tag domain. Though the technique we present in this paper is in no way specific to adversarial environments, we will adopt the terminology of laser tag throughout for simplicity of discussion, referring to the tracking robots as the teammates, and those robots being tracked as the opponents.

2. TRACKING UNDER OCCLUSION

2.1 State Representation

The main contribution of this paper is a particle filter algorithm for tracking the location of a large number of opponents under prolonged periods of occlusion. The teammates make use of sensor information to track their opponents. Let $y_{1:t} = y_1, y_2, \ldots, y_t$ denote the teammates' measurements up to time t, and $u_{1:t}$ their control inputs. In our implemented system, sensor measurements consist of range readings at a known set of bearings from each teammate produced by SICK laser range finders, but with appropriate modifications to the sensor model the readings could come from other perception systems such as vision or sonar. We will write y_{it} for the ith individual reading.

As is common in the literature on tracking [3], our approach maintains a posterior probability distribution over the state, which we denote x_t, at time t: $P(x_t \mid y_{1:t}, u_{1:t})$. Note that we will sometimes use the notation $P(X_t = x_t \mid y_{1:t}, u_{1:t})$ which has the same meaning. For the laser tag problem, the state variables consist of the positions of all of the agents. These variables can be divided into two qualitatively different sets: the positions of the teammates and the positions of the opponents.

More formally, our approach divides the state x_t into two pieces (r_t, s_t). The first piece, r_t, contains the positions of our own teammates. The second piece, s_t, contains the positions of the opponent robots. We divide s_t into $s_t^1, s_t^2, \ldots, s_t^M$ for the M individual opponents and let $s_{1:t}$ represent the complete history of opponent positions. In addition to the state variables we will introduce data association variables $z_{1:t}$. The data association variables z_t are actually a collection of binary variables $z_t^{i\to j}$, each of which indicate whether a sensor reading y_{it} is associated with an opponent j. We further introduce a special set of association variables, $z_t^{i\to 0}$, to specify whether a sensor reading is associated to the map. Each reading is associated to exactly one object in our world. We can express this constraint formally as $\sum_{j=0}^{M} z_t^{i\to j} = 1$. We will sometimes use the special notation $z_t^{\to m}$ to represent the set of all association variables that indicate some sensor reading associates to opponent m, for example $z_t^{\to 0}$ represents the set of all association variables that indicate some sensor reading associates to the map. We will never represent $z_{1:t}$ explicitly, but we will need to reason about it in the derivation of the state-tracking equations below.

With the above notation, our tracking problem is to estimate the posterior

$$P(r_t, s_t, z_t \mid y_{1:t}, u_{1:t}, z_{1:t-1})$$

Unfortunately, maintaining this distribution can be difficult and computationally expensive due to complicated multi-modal dependencies between state variables. The main cause of these problems is that the teammates' sensor views of the opponents are frequently occluded. This occlusion makes it difficult to determine which opponent is which, and it can also put sharp edges and multiple modes into our posterior distributions due to our inability to see around corners and through walls. In order to make tracking computationally feasible, we will work through a series of three factorizations which separate out various tractable pieces of the problem. These steps are *conditioning on teammates*, *factoring by role*, and *factoring observation and motion models*. We will describe each step in detail below.

2.2 Conditioning On Teammates

The tracking problem is significantly simplified by factoring the posterior probability

$$\begin{aligned} & P(r_t, s_t, z_t \mid y_{1:t}, u_{1:t}, z_{1:t-1}) \\ = \quad & P(r_t, z_t^{\to 0} \mid y_{1:t}, u_{1:t}, z_{1:t-1}) \\ & P(s_t, z_t^{\to\{1\ldots M\}} \mid r_t, y_{1:t}, u_{1:t}, z_{1:t-1}, z_t^{\to 0}) \end{aligned} \tag{1}$$

This factorization is an application of Bayes' rule. The first term in the resulting expression encapsulates the problem of self localization, representing only a distribution over the teammates and the association variables that pertain to the map. The second term is the posterior over opponent positions. Now we can solve these two problems independently. This is an important simplification because the self-localization problem has been studied extensively [9] and we can now apply standard techniques. The constraint, mentioned previously, that each sensor reading is associated to exactly one source introduces a slight difficulty in the implementation of this factorization. We will revisit this issue in Section 4.1.

In our domain, a particle-filter-based localization technique [17, 33] nearly always produces unimodal, high-accuracy position estimates for robots on our own team, so long as we have a reasonable idea of their initial positions. This motivates us to approximate the positions of our teammates by point estimators located at the most likely position for each robot. Such an approximation is only appropriate when the uncertainty for the positions of our robots is small, as it precludes us from using events such as robots observing each other for the self-localization part of our tracking problem [9]. In practice the uncertainty of our own robots' positions is usually extremely small, so this approximation is valid.

2.3 Data Association Under Occlusion

We have now reduced the tracking problem to estimating

$$P(s_t, z_t^{\to\{1\ldots M\}} \mid r_t, y_{1:t}, u_{1:t}, z_{1:t-1}, z_t^{\to 0})$$

We know the observations $y_{1:t}$ and controls $u_{1:t}$, and the robot positions r_t and map associations $z_t^{\to 0}$ are given by our robots' self-localizers. Unfortunately the remaining association variables $z_t^{\to\{1\ldots M\}}$ are more problematic.

Data association uncertainty causes two unpleasant effects in our posterior distribution, as illustrated in Figure 2a. First, each plausible data association hypothesis can cause a separate mode in our

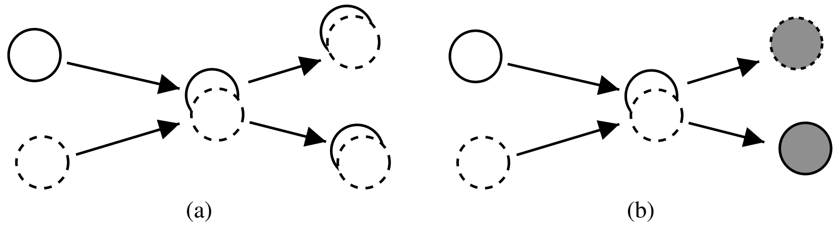

Figure 2: (a) We can confuse two tracked objects if they move close together and then apart. This confusion manifests in two ways: multi-modal distributions for each object, and negative correlation between the positions of the two objects. (b) If we allocate new roles to represent "the robot which went left" and "the robot which went right," we have no data association problem and only unimodal posterior beliefs.

belief for the location of opponent actors. Because the number of plausible data associations often grows exponentially with time, the result is a highly multi-modal belief. Second, these modes contain cross-robot correlations: if we are confused whether a particular object we are tracking is opponent 2 or opponent 3, then finding out (through some other evidence) that it is opponent 2 causes us to believe opponent 3 is elsewhere. This is a common characteristic of data association problems [3].

In many tracking problems, it suffices to keep track of the single most probable data association hypothesis. For example, Dissanayake and colleagues [6] describe a Kalman filter tracker which uses maximum-likelihood data association to keep track of hundreds of landmarks. In their problem, however, landmarks are fairly well separated compared to the amount of uncertainty in their positions, which means that the most likely data association usually accounts for most of the posterior probability mass.

In the laser tag problem, unfortunately, we are not so lucky. Because opponents may be occluded for sizable intervals, their uncertainty areas can grow quite large and will quickly begin to overlap significantly with each other. For example, our sensor log might show two opponents entering an initially empty room and, after a pause, one opponent exiting the room. In this case, we cannot determine the true identity of the exiting opponent; all we know is that it is one of the two which went in, and that one other must still be inside the room.

This problem has important ramifications. If we were to attempt to keep track of the exact posterior distribution for all actors, we would quickly be overwhelmed with the number of plausible data association hypotheses. However, keeping track of only the most likely hypothesis would result in suboptimal behavior: for example, we might accidentally relabel two opponents as each other. If we have already tagged one of the two, that mistake will cause us to let the other pass freely. To our knowledge, existing tracking and data association methods are unable to cope with such situations.

This same example leads to another important and related insight: occlusion introduces structure into our posterior. There are two possible data association decisions we can make as we see an opponent exit the room: either the first opponent to enter the room is now exiting, or the second is. This is a difficult decision exactly because both associations are plausible. The underlying cause of this situation is that we cannot see inside the room. If an opponent enters and becomes occluded for a sizable interval then we know little about its position besides that it is in the room, and that statement holds for every robot we have seen enter. Notice that a symmetry has been introduced into our posterior. The occlusion afforded by the room, given sufficient time, will cause our belief for every opponent inside to become identical. In order to exploit this symmetry to allow us to track very large numbers of opponents efficiently and, at the same time, drastically simplify our data association problem, we now factor our posterior into roles.

2.4 Factoring by Role

Roles represent classes of opponents that share similar posterior distributions. For example one role might represent the position of the opponent we just saw come out of a room, without reference to its exact identity. The definition of roles makes it possible to track multiple opponents under severe data association problems arising from the nature of the occlusions. In particular, we can allocate or merge roles dynamically as we gain or lose information about the locations of opponents.

This sort of role-based (also called *deictic* [2]) representation allows us to collapse together many different symmetric modes of our posterior distribution, as indicated in Figure 2b.[1] Roles also enable us to overcome the the data association problem: we can define the roles so that we are certain which role should be associated with each sensor reading, regardless of the identity of a specific opponent. That means that that there will be no correlation between the positions of different roles, and so we can factor our belief and track each role separately. It also means we can track arbitrarily many opponents as long as they fall into a compact set of roles, as will often happen if there is pervasive occlusion.

In our model roles mean that several opponents may share a single piece of the opponent state s_t. To accomplish this, we introduce a function $d(\ell)$ that maps opponent indices into role indices. Specifically it takes an opponent index ℓ and returns the index of that opponent's role. In general several opponents will fall into a single role. We define the role index of each opponent in a role as the opponent index of the first opponent in the role. For example, suppose there are five opponents with identity indexes 1-5 and they are divided into two roles, $\{1, 3, 5\}$ and $\{2, 4\}$. In this case $d(1) = d(3) = d(5) = 1$ and $d(2) = d(4) = 2$. This particular method for assigning indices to roles is arbitrary, but will afford us some notational convenience. We will denote the set of active roles in our system Γ and the number of opponents assigned to a particular role $\gamma \in \Gamma$ as $\#(\gamma)$.

The role-based representation has an important impact on the posterior estimation problem. Mathematically, it allows us to simplify our belief. It is well known that we can approximately fac-

[1] In practice, we rarely need to worry about confusion of the sort illustrated in Figure 2. Much more frequent is the type of confusion illustrated in Figure 3 below, where two opponents leave our sensor range in the same direction and quickly become indistinguishable. But the same principle applies in either case: if we try to track the robots individually we wind up with multi-modal posteriors and global correlations, while if we reason about roles as described in this section the posterior becomes much easier to track.

tor the problem of tracking several objects using positive information into several independent tracking problems with one object each [26], therefore we can rewrite our posterior as:

$$P(s_t, z_t^{\to\{1\dots M\}} \mid r_t, y_{1:t}, u_{1:t}, z_{1:t-1}, z_t^{\to 0})$$
$$= \prod_{\ell=1}^{M} P(s_t^\ell, z_t^{\to\ell} \mid r_{1:t}, y_{1:t}, u_{1:t}, z_{1:t-1}, z_t^{\to 0})$$

Now we factor our posterior by role. We simply rewrite the posterior replacing all the opponent indices in the equation with the function $d(\ell)$ which maps them into roles.

$$\approx \prod_{\ell=1}^{M} P(S_t^{d(\ell)} = s_t^\ell, z_t^{\to d(\ell)} = z_t^\ell \mid r_{1:t}, y_{1:t}, u_{1:t}, z_{1:t-1}, z_t^{\to 0}) \quad (2)$$

Notice that we have chosen our roles so that we can track them independently. Note that many of the terms in the above equation are similar. This is because we are tracking roles and not individuals opponents and in general there are far fewer roles than individuals. The advantage of this factorization lies in the fact that it makes tracking highly efficient. The fact that there are far fewer roles than individuals leads directly to fewer particle filter tracks in our implementation. In contrast, the common formulation of the tracking problem requires us to maintain exponentially many hypotheses, which clearly is impractical in the presence of many opponents.

2.5 Factoring Sensor and Motion Models

We have now given a factorization that will allow us to track many objects efficiently. However, we have not yet shown that this factorization is maintained over time. With joint motion and observation models, the positions of different roles might become dependent whenever we moved or observed, destroying the factorization which we obtained in the previous steps. This is *not* the case for motion and observation models that model each role independently. As discussed in this section, both the motion model and the measurement model are indeed independent for different roles, validating the factorization proposed in the previous section.

The independence of the motion model for teammates is straightforward. Each of our teammates moves according to local motion commands; the noise in motion is independent for each of the robots. Similarly, it is natural to assume that the opponents move independently of each other, with independent random variables characterizing their next state transition functions. Clearly, the latter independence assumption might not actually hold: for example, it would be violated if we knew that opponents tended to move in groups. In practice, though, assuming independence in opponent motion is safe in that it establishes a *worst case* motion model.

Independence also holds for measurements, as discussed at length in [19]. In particular, there are two types of information we can get from our sensors, *positive* and *negative*. Positive information tells us where an opponent actor *is*; we receive positive information by associating a sensor reading i to a role j. Negative information tells us where an opponent is *not*; we receive negative information when our sensor beams pass through a space without detecting an opponent. Negative information produces complicated posteriors with sharp edges because our visibility region is distorted by occlusion from static obstacles in the map. Incorporating negative information also tends to introduce multiple modes in our posterior because a teammate's field of view may cut out the middle of our posterior over an opponents position. In fact, negative information can increase the variance of our belief distribution (although it always reduces entropy). Most position-tracking systems can only incorporate positive information [3]. If the tracked objects are only rarely occluded, positive information is usually sufficient to achieve good tracking. In laser tag, however, the majority of the information we receive is negative, and so it is critical to take advantage of negative information in our tracking system. For example, if our system observes an opponent enter a room with only one exit, it is imperative to keep track of the fact that it remains in the room until our robots see it leave (or until our robots lose sight of the exit).

It is well known that we can approximately factor the problem of tracking several objects using positive information into several independent tracking problems with one object each [26]. For our system, though, we make the novel observation that we can also factor the effects of negative information. The observations enter into the tracking problem through the observation likelihood $P(y_t \mid r_t, s_t, z_t)$. Since each observation is conditionally independent once we know positions and data associations, we have

$$P(y_t \mid r_t, s_t, z_t) = \prod_i P(y_{it} \mid r_t, s_t, z_t)$$

Now, suppose observation i is associated to role k. In this case, the probability of observation i is a product of two terms: the probability of generating y_{it} given that the sensor beam reached role k, and the probability that the beam reached role k. The latter term in turn factors into the probability that the beam avoided interception by role 1, times the probability that the beam avoided interception by role 2, and so forth for all roles $j \neq k$. In other words, we can write

$$P(y_t \mid r_t, s_t, z_t) = \prod_i \Bigg[P(y_{it} \mid r_t, s_t^k, z_t^{i\to k} = 1) \prod_{j \in \Gamma \neq k} P(y_{it} \mid r_t, s_t^j, z_t^{i \to j} = 0)^{\#(j)} \Bigg] \quad (3)$$

which is factorized so that each term depends only on the belief for only a single role. Thus, sensor measurements maintain the conditional independence of our role estimates, and so our factorized representation remains valid as we incorporate sensor information.

2.6 Summary of Tracking Algorithm

Combining equations (1) and (2), our final factorization of the belief state is

$$P(r_t, s_t, z_t \mid y_{1:t}, u_{1:t}, z_{1:t-1}) = P(r_t, z_t^{\to 0} \mid y_{1:t}, u_{1:t}, z_{1:t-1}) \prod_{\ell=1}^{M} P(S_t^{d(\ell)} = s_t^\ell, z_t^{\to d(\ell)} = z_t^\ell \mid r_{1:t}, y_{1:t}, u_{1:t}, z_{1:t-1}, z_t^{\to 0}) \quad (4)$$

Equation (3) shows that this factorization is preserved across evidence updates, and a similar argument shows that it is preserved across motion updates.

In contrast to the naive approaches of tracking each opponent separately or all opponents together—neither of which is computationally feasible for problems with pervasive occlusion—our factorization represents an efficient and accurate way to compute our belief state after any sequence of observations and actions. The reason for this efficiency is that we have separated the overall belief-tracking problem into a number of smaller tracking problems.

3. ROLE ALLOCATION

In order to fully specify our tracking algorithm, the only remaining step is to describe how we allocate and merge roles in order to maintain independence. We begin with a single role which represents all of the robots on the opposing team. This role's position is initialized to a uniform distribution over free space. As we receive new observations, there are two types of decisions we must make: when to split a role into two, and when to merge two roles into one.

Role splitting is driven by positive information: if a positive observation's maximum likelihood association is to a role j that represents more than one opponent, our approach splits off a copy of role j (call this copy k). Role k starts out identical to role j in every way except except it represents only one robot. (We of course must also decrement the number of opponents represented by role j.) Now we can incorporate the new observation into role k. When doing so, k's position uncertainty will become smaller than j's. In this way, future observations in similar locations will tend to associate with k instead of j, causing role k to track the newly-observed robot while j represents the position of the remaining unseen robots.

Role merging is a more expensive operation. After every T_{merge} time steps, our approach checks each role j against each other role k to see if their position distributions are very similar. If they are, we can no longer tell the two roles apart, and so we can merge them into one. This test is quadratic in the number of roles, but since it tends to keep the number of roles small the expense is usually worthwhile. (In any case it is much lower than the exponential costs associated with tracking multiple modes for each individual opponent.)

For our measure of similarity of distributions, we use a grid-based symmetric KL-divergence. This measure places a coarse grid over the map and estimates the probabilities p_{ij} and p_{ik} that roles j and k assign to each grid cell i (using Laplace smoothing to ensure that no grid cell is assigned zero probability). Based on the frequency counts in this grid, our approach calculates symmetric KL-divergence:

$$D_{KL}(p_j, p_k) = \sum_i \left[p_{ij} \ln \frac{p_{ij}}{p_{ik}} + p_{ik} \ln \frac{p_{ik}}{p_{ij}} \right]$$

between two different role posteriors. If this divergence is smaller than a cutoff, the roles j and k are merged into a single role. In the laser tag domain, the grid is two-dimensional.

When we merge two roles, the merged position distribution becomes the average of the two original position distributions before the merging operation (which is a good approximation since these two distributions were just determined to be nearly the same). This new role is initialized to represent the sum of the number of opponents represented by the original two.

4. EXPERIMENTAL RESULTS

In order to better understand the strengths of our system we performed a number of experiments, both in simulation and on real robots. First, we evaluated the utility of dynamically merging roles and the impact of that feature on scaling performance. Second, we systematically explored the impact of our tracking algorithm on our ability to find and tag opponents in the laser tag setting. Finally, we verified that our system performs well in large scale real world environments on real robots. In each of these experiments we found that our proposed system exhibits excellent performance. The following sections will explain several implementation details of our algorithm and then treat each experiment in detail.

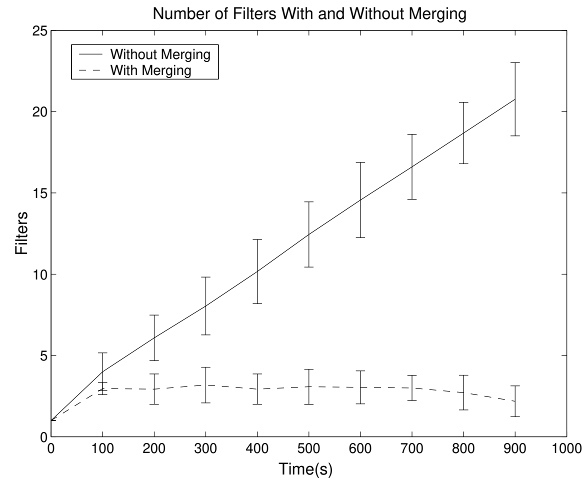

Figure 4: In this experiment an observer watches another robot move through a hall. Merging allows us to maintain a near constant number of filters, since there are only a constant number of beliefs in this system. Without merging the number of filters grows linearly.

4.1 Implementation Details

For completeness, we now review the details of approximations to the terms of equation (4) made by our implementation. Our approach approximates the first term $P(r_t, z_t^{\rightarrow 0} \mid y_{1:t}, u_{1:t}, z_{1:t-1})$ in equation (4) with a vector of maximum-likelihood estimates returned by individual self-localizers, one for each team member. A subtlety of this first term is that our implementation picks $z_t^{\rightarrow 0}$ before the $z_t^{\rightarrow 1 \ldots M}$ are chosen. This is an approximation because of the constraint $\sum_{j=0}^{M} z_t^{i \rightarrow j} = 1$ discussed in Section 2.1. This approximation has little practical effect because our localizer is quite good at identifying map features.

In our implementation we represent each term inside the product in equation (4) with a separate particle filter tracker. This representation is inexact since particle filters are Monte Carlo algorithms, but it is asymptotically correct as we increase the number of particles. Since each individual particle filter only has to track a single role, we have observed that we do not need too many particles in practice to get good tracking performance.

4.2 Dynamic Role Assignment

In order to demonstrate the effectiveness of dynamically adding and merging roles we placed, in simulation, an observer robot in a doorway looking into a hall. We then moved a sequence of opponents through the hallway past the door at random intervals. The setting of the experiment is shown in Figure 3: the light-filled bold circle represents the observer robot, with a tic-mark showing orientation, and the dark-filled bold circle is the opponent. Each row of the figure represents the state of all active particle filters in the system at different times during a pass of a single opponent. In this example we are allowing roles to dynamically merge over time, so the number of active filters can decrease. If we do not have this functionality the number of active filters is strictly increasing.

To understand the utility of roles in tracking, we ran the system with and without dynamic role merging. In each case we ran the entire experiment 25 times, with a single run consisting of approximately 20 opponents moving in sequence past the observer robot. Figure 4 shows the average number of active particle filters in the system in both cases. Without dynamic merging, the system keeps

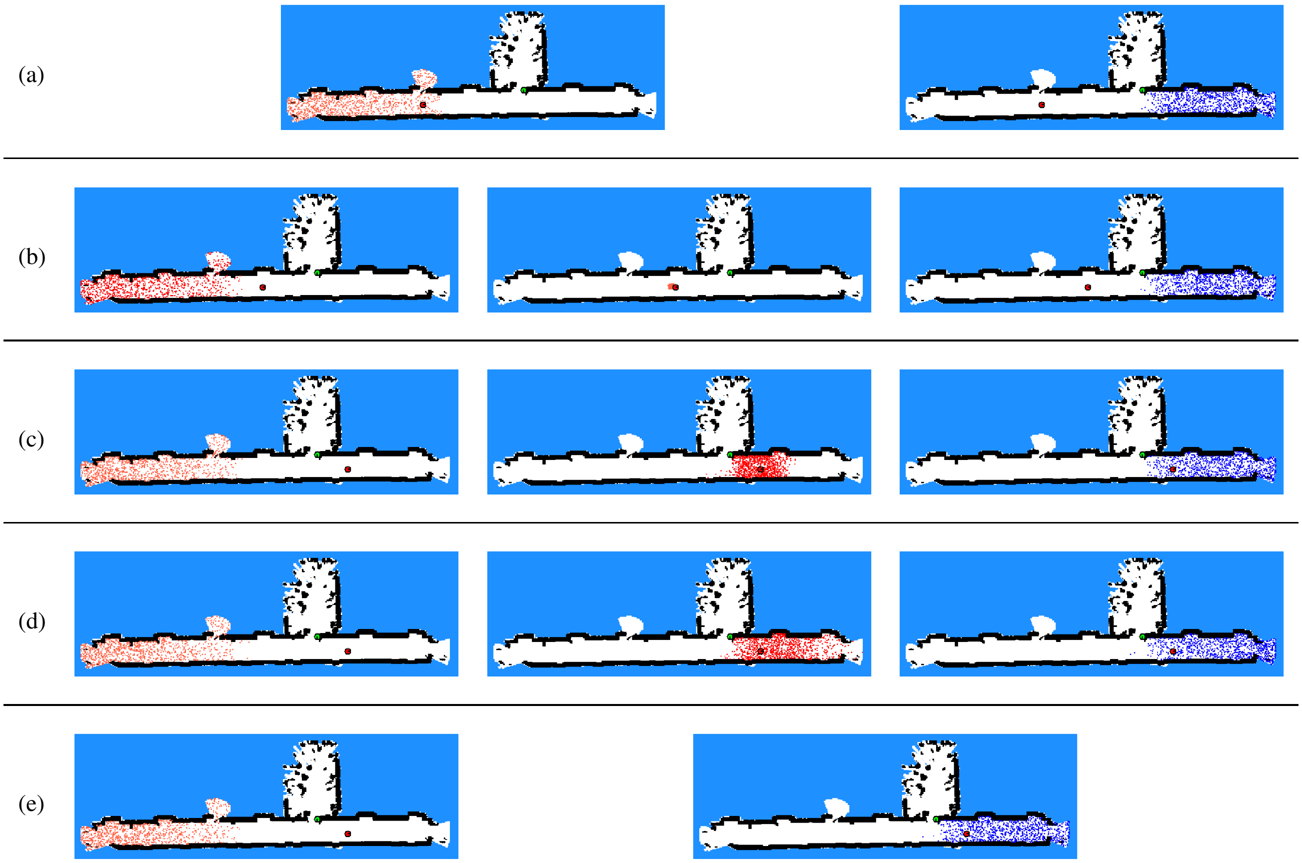

Figure 3: Each row represents the state of all the filters in the system at a given time. (a) Before splitting there are only two filters. (b) Just after splitting there are three filters. (c) The robot has moved down the hall, out of sight of the observer and the distribution that was assigned to it has begun to spread. (d) Distribution 2 is becoming similar to distribution 3. (e) Distributions 2 and 3 were deemed similar enough to merge.

track of all the robots independently, despite the fact that their distributions are virtually identical. Thus, the number of active particle filters grows with time. With dynamic merging the number of filters remains essentially constant as similar distributions are merged.

4.3 Tracking

We explored the utility of our tracking system in the laser tag domain by implementing two planners for locating and tagging opponents. The first of these planners uses our tracker to model the location of the opponents. The second, which we will refer to as the baseline planner, is similar to the first except it does not use a tracker.

In both planners point to point navigation is implemented through standard path planning algorithms [21]. For multi-robot coordination of our teammates, our tracker-based planner uses an heuristic approach where the teammates greedily attempt to maximize information gain, but they do so one at a time so that they can take previous teammates decisions into account. The baseline planner explores randomly, but uses the same technique of having the teammates chose actions one at a time in an attempt to avoid redundancy.

More specifically the coordination strategy in both planners works as follows. Every T_{replan} seconds each of the teammates chooses a point to move toward. These destination points are chosen from a coarse three-dimensional grid, (x, y, θ), of points laid over the map. Here θ is the robot orientation, which is important because each robot's laser sensor covers only the area in front of it. In both planners, teammates compute values for each of the grid points, with higher values corresponding to more desirable locations. However, the values are computed differently for the different planners.

In the tracker-based planner, the values of the destination points are a function of several factors including the travel time required to reach them, the nuber of hypothisized opponent positions that would be observed were the teammate there, the degree to which those hypotheses would be centered in the teammate's sensor field, and whether or not other teammates have already chosen nearby destinations. The degree to which each of these factors influence planning is controlled by empirically chosen weights. In the baseline planner the only contributing factors are the travel time and whether or not other teammates have already chosen nearby points.

Coordination strategies similar to our tracker-based solution have been found to be highly effective in the context of coordinated multi-robot exploration of static environments [5, 31, 34]. We chose the baseline planner to represent the class of behavior-based strategies which do not attempt to track the true posterior over opponent positions. We believe that, within the class of memoryless strategies, this baseline planner is a good performer; and, since it is very similar in architecture to the tracker based planner, it provides a good basis for comparison.

We compared these two strategies over 100 runs in simulation. A single run consists of all robots (two teammates and one opponent) starting at random positions, and the teammates searching until the opponent is tagged. If our solution, with tracking, allows us to tag

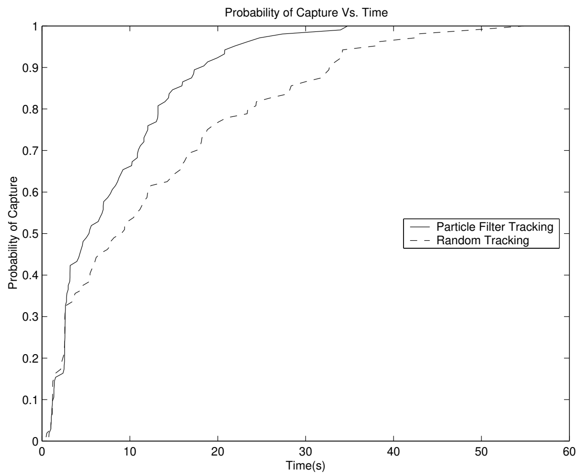

Figure 5: The state tracking in our system allows it to find the opponent consistently faster than a baseline system without the benefit of tracking.

the evader more quickly than the baseline system, we can conclude that our tracking algorithm helps the teammates find the opponent. If the tracking approach does not yield superior performance then we will conclude that state estimation is not useful in this domain, that the environment we are searching is too small and simple, or that the planner we are using is not capable of taking advantage of tracking information. Figure 5 plots the probability of capture versus time for the two approaches. In other words the graph shows the fraction of runs that completed by each time step. Our system consistently outperforms the baseline, demonstrating the value of our state tracking algorithm.

4.4 Complete System

Figure 6 shows a run of the complete system on physical robots. There are two teammates, a Pioneer I and a Pioneer II, and one opponent, built on a modified Scout base. Each robot carries a Pentium-class computer which is used to run components of the Carmen software suite [17]. These components provide each robot with localization and point-to-point navigation in pre-built maps. In order to control the teammates, we use the tracker based planning strategy described in Section 4.3.

In figure 6, panel (a) shows the initial conditions. The two teammates begin near the center of the hallway. They are represented by filled circles with a dark outline and a radial line indicating orientation. The position of the opponent is shown at the far left of the hall (also a filled circle, but with a slightly different shading). The position of the opponent is not used by the teammates for planning, but only for distinguishing it from ancillary objects, such as people that might walk by during an experiment. In (b), the teammates begin moving toward the destination points selected by the coordination technique (represented by unfilled circles with a light outline and a radial line to indicate orientation). Note that they have chosen to move in separate directions and have also cleared most of the hypotheses in the room above the corridor. Moving to (c), the right end of the hall has been nearly cleared of hypotheses, meaning it is very unlikely the opponent is hiding there, so both teammates decide to move toward the left of the corridor. In (d) the opponent is finally located and tagged by one of the teammates. At this point the teammates label the opponent as having been tagged and ignore it in their planning, though they still track it to avoid confusing it

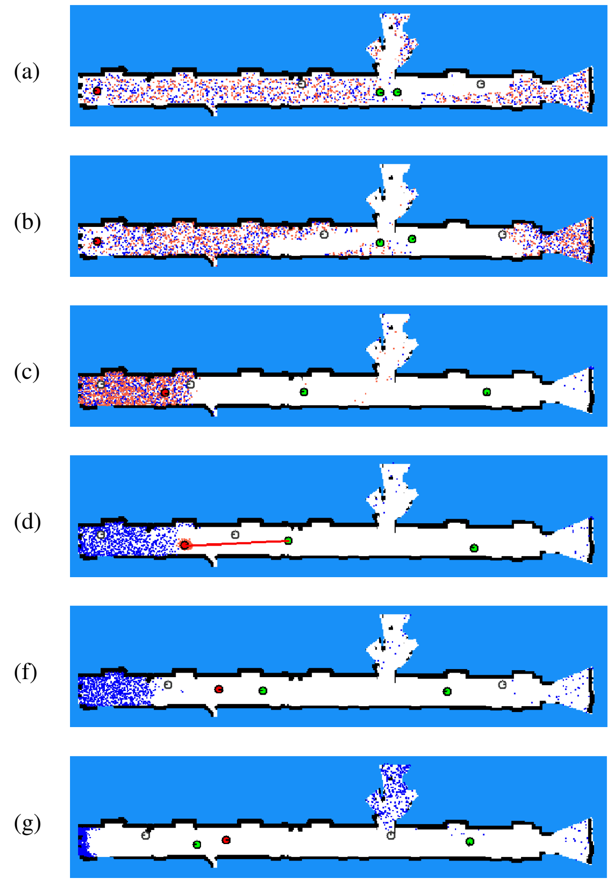

Figure 6: A complete run of the full system on real robots.

with another opponent. In (f) and (g) the teammates continue to search the map since they have been told that there are potentially two opponents in the world and they have only tagged one.

We have run our system in several environments besides the one detailed above. Figure 7 shows maps with example posteriors from two other environments, as well as a photo of the robots in the environment described above. In all these environments the system performed well, directing the teammates to search through the map efficiently until they found the target. During these runs the system displayed intelligent behaviors such as splitting the robots up to explore branching corridors and keeping the robots together to cover wide areas more efficiently without letting the opponent past.

5. RELATED WORK

Multi-robot information gathering has been addressed by many researchers. The classical setting involves a team of robots locating stationary objects in an unknown environment [35, 10, 16, 23]. Most existing work in this field involves behavior-based strategies, in which the search is carried out through randomized motion. Coordination is often achieved through behaviors that maximize the distance between adjacent robots. Research in this field has predominantly focused on static environments [1], although some notable success has been reported for environments with dynamic objects [29]. However, randomized search is limited in that it relies on chance to find objects.

Techniques that maintain environment models during search have been studied extensively in the field of multi-robot mapping [5, 20,

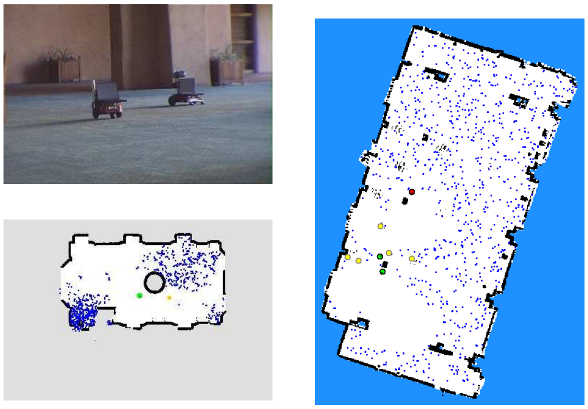

Figure 7: Environments in which our robots have played laser tag.

31]. These approaches apply to static environments in which the objects being tracked do not move. The tracking literature has also thoroughly addressed the issue of tracking moving objects [3, 26, 15]. This work has recently been extended to mobile robots [11, 13, 18, 30] and distributed sensor systems [20, 24]. While these approaches work well in cases where the objects of interest lie within sensors' reach (with possible brief periods of occlusion), they do not address the type of long-term occlusion found in laser tag.

6. DISCUSSION AND CONCLUSION

We have introduced a system for playing multi-robot laser tag. Laser tag presents an interesting opportunity for research because of its real-time, dynamic nature and because it leads to complex non-Gaussian beliefs. We conjecture that the research issues that result are characteristic of a much broader range of multi-robot application domains, ranging from personal service robots (e.g., tour-guide robots [4, 22, 32]) to the coordination of unmanned air vehicles on reconnaissance missions.

Our system works primarily because of a new particle-filter based algorithm that allows us to track arbitrary numbers of opponents in the presence of pervasive occlusion. This is accomplished through a factorization that allows us to track opponents in terms of roles instead of individual identities, taking advantage of the structure induced in our posterior by the occlusion.

Our research opens up many opportunities for future work. An avenue of particular interest is sophisticated planning. One possible approach to this problem is to use belief compression [28] to compactly represent the belief state and use MDP planning in the compressed space. Another direction to explore is creating a more sophisticated opponent model to increase performance against smarter adversaries. A third interesting extension is to augment the tracker to handle some types of correlations among adversaries; this would allow us to, for example, build play-books of opponent tactics and use knowledge of these plays to infer something about occluded opponents given the known positions of others.

We believe that the laser tag problem is highly interesting for multi-agent mobile robotics research. Just like robotic soccer, it involves fast-moving entities; however, the nature of the sensor data (specifically, pervasive occlusion) poses new challenges in robot perception. As this paper suggests, these problems can be addressed through a novel tracking technique that represents opponents by role instead of individual identity.

7. REFERENCES

[1] T. Balch, G. Boone, T. Collins, H. Forbes, D. MacKenzie, and J. C. Santamaria. Io, ganymede and callisto – a multiagent robot janitorial team. *AAAI Magazine*, 16(1), 1995.

[2] Dana H. Ballard, Mary M. Hayhoe, Polly K. Pook, and Rajesh P. N. Rao. Deictic codes for the embodiment of cognition. TR NRL95.1, 1996.

[3] Y. Bar-Shalom and T. E. Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[4] W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. Experiences with an interactive museum tour-guide robot. *Artificial Intelligence*, 114(1-2), 1999.

[5] W. Burgard, D. Fox, M. Moors, R. Simmons, and S. Thrun. Collaborative multi-robot exploration. *Proc. ICRA-00*.

[6] G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localisation and map building (SLAM) problem. *IEEE Transactions of Robotics and Automation*, 2001.

[7] A. Doucet, J.F.G. de Freitas, and N.J. Gordon, editors. *Sequential Monte Carlo Methods In Practice*. Springer, 2001.

[8] Arnaud Doucet, Nando de Freitas, Kevin Murphy, and Stuart Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. *Proc. UAI-00*

[9] D. Fox, W. Burgard, H. Kruppa, and S. Thrun. A probabilistic approach to collaborative multi-robot localization. *Autonomous Robots*, 8(3), 2000.

[10] D. Goldberg and M.J. Matarić. Robust behavior-based control for distributed multi-robot collection tasks. TR IRIS-00-387, USC, 2000.

[11] H. H. Gonzalez-Banos and J. C. Latombe. Navigation strategies for exploring indoor environments. *International Journal of Robotics Research*, 2002.

[12] H. Kitano, editor. *Proc. RoboCup-97*, Springer.

[13] S.M. LaValle, H. Gonzalez-Banos, C. Becker, and J.C. Latombe. Motion strategies for maintaining visibility of a moving target. *Proc. ICRA-97*.

[14] J. Liu and R. Chen. Sequential monte carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93, 1998.

[15] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. *Proc. ICCV-99*.

[16] M.J. Matarić. Issues and approaches in the design of collective autonomous agents. *Robotics and Autonomous Systems*, 16, 1995.

[17] M. Montemerlo, N. Roy, and S. Thrun. Carnegie mellon robot navigation toolkit. Software package, download at www.cs.cmu.edu/∼carmen, 2002.

[18] M. Montemerlo, W. Whittaker, and S. Thrun. Conditional particle filters for simultaneous mobile robot localization and people-tracking. *Proc. ICRA-02*.

[19] K. Murphy. Bayesian map learning in dynamic environments. *Proc. NIPS-99*.

[20] E. Nettleton, H. Durrant-Whyte, P. Gibbens, and A. Goktoğan. Multiple platform localisation and map building. In *Sensor Fusion and Decentralised Control in Robotic Systems III*, 4196, 2000.

[21] N.J. Nilsson. *Principles of Artificial Intelligence*. Springer, 1982.

[22] I. Nourbakhsh, J. Bobenage, S. Grange, R. Lutz, R. Meyer, and A. Soto. An affective mobile robot with a full-time job. *Artificial Intelligence*, 114(1–2), 1999.

[23] L. E. Parker. On the design of behavior-based multi-robot teams. *Journal of Advanced Robotics*, 10(6), 1996.

[24] H. Pasula, S. Russell, M. Ostland, and Y. Ritov. Tracking many objects with many sensors. *Proc. IJCAI-99*.

[25] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. M. Kaufmann, 1988.

[26] D.B. Reid. An algorithm for tracking multiple targets. *IEEE Transactions on Aerospace and Electronic Systems*, AC-24, 1979.

[27] http://robotag.com/

[28] Nicholas Roy and Geoffrey Gordon. Exponential family PCA for belief compression in POMDPs. *Proc. NIPS-02*.

[29] R. Sargent, B. Bailey, C. Witty, and A. Wright. Dynamic object capture using fast vision tracking. *AI Magazine*, 18(1), 1997.

[30] D. Schulz, W. Burgard, D. Fox, and A. Cremers. Tracking multiple moving targets with a mobile robot using particles filters and statistical data association. *Proc. ICRA-01*.

[31] R. Simmons, D. Apfelbaum, W. Burgard, M. Fox, D. an Moors, S. Thrun, and H. Younes. Coordination for multi-robot exploration and mapping. *Proc. AAAI-00*.

[32] S. Thrun, M. Beetz, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Hähnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Probabilistic algorithms and the interactive museum tour-guide robot Minerva. *International Journal of Robotics Research*, 19(11), 2000.

[33] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial Intelligence*, 128(1-2), 2000.

[34] B. Yamauchi, P. Langley, A.C. Schultz, J. Grefenstette, and W. Adams. Magellan: An integrated adaptive architecture for mobile robots. TR 98-2, ISLE, 1998.

[35] Cao Y.U., Fukunaga A.S., and Kahng A.B. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4, 1997.