An MCMC-based Particle Filter for Tracking Multiple Interacting Targets

Zia Khan, Tucker Balch, and Frank Dellaert
College of Computing
Georgia Institute of Technology, Atlanta, GA
{zkhan,tucker,frank}@cc.gatech.edu

Technical Report number GIT-GVU-03-35
October 2003

Abstract

We describe a Markov chain Monte Carlo based particle filter that effectively deals with *interacting* targets, i.e., targets that are influenced by the proximity and/or behavior of other targets. Such interactions cause problems for traditional approaches to the data association problem. In response, we developed a joint tracker that includes a more sophisticated motion model to maintain the identity of targets throughout an interaction, drastically reducing tracker failures. The paper presents two main contributions: (1) we show how a Markov random field (MRF) motion prior, built on the fly at each time step, can substantially improve tracking when targets interact, and (2) we show how this can be done efficiently using Markov chain Monte Carlo (MCMC) sampling. We prove that incorporating an MRF to model interactions is equivalent to adding an additional *interaction factor* to the importance weights in a joint particle filter. Since a joint particle filter suffers from exponential complexity in the number of tracked targets, we replace the traditional importance sampling step in the particle filter with an MCMC sampling step. The resulting filter deals efficiently and effectively with complicated interactions when targets approach each other. We present both qualitative and quantitative results to substantiate the claims made in the paper, including a large scale experiment on a video-sequence of over 10,000 frames in length.

1 Introduction

This work is concerned with the problem of tracking multiple interacting targets. Our objective is to obtain a record of the trajectories of targets over time, and to maintain correct, unique identification of each target throughout. Tracking multiple identical targets becomes challenging when the targets pass close to one another or merge.

The classical multi-target tracking literature approaches this problem by performing a data-association step after a detection step. Most notably, the multiple hypothesis tracker [26] and the joint probabilistic data association filter (JPDAF) [4, 12] are influential algorithms in this class. These multi-target tracking algorithms have been used extensively in the context of computer vision. Some examples are the use of nearest neighbor tracking in [9], the multiple hypothesis tracker in [7], and the JPDAF in [25]. Recently, a particle filtering version of the JPDAF has been proposed in [27].

In this paper we address the problem of *interacting* targets, which causes problems for traditional approaches. Dealing appropriately with this problem has important implications for vision-based tracking of animals, and is generally applicable to any situation where many interacting targets need to be tracked over time. Visual animal tracking is not an artificial task: it has countless applications in biology and medicine. In addition, our long term research goals involve the analysis of multi-agent system behavior in general, with social insects as a model [3]. The domain offers many challenges that are quite different from the typical radar tracking domain in which most multi-target tracking algorithms are evaluated.

In contrast to traditional methods, our approach relies on the use of a more capable motion model, one that is able to adequately describe target behavior throughout an interaction event. The basic assumption on which all established data-association methods rely is that targets maintain their behavior before and after the targets visually merge. However, consider the example in Figure 1, which shows 20 ants being tracked in a small arena. In this case, the targets do *not* behave independently: whenever one ant encounters another, some amount of interaction takes place, and the behavior of a given ant before and after an interaction can be quite different. The approach we propose is to have the motion model reflect this additional complexity of the target behavior.

The first contribution of this paper is to show how a Markov random field motion prior, built on the fly at each time step, can adequately model these interactions and defeat these failure modes. Our approach is based on the well known particle filter [15, 18, 6, 8], a multi-hypothesis tracker that uses a set of weighted particles to approximate a density function corresponding to the probability of the location of the target given observations over time. The standard particle filter weights particles based on a likelihood score, and then propagates these weighted particles according to a motion model. Simply running multiple particle filters, however, is not a viable option: whenever targets pass close to one another, the target with the best likelihood score typically “hijacks” the filters of nearby targets, as is illustrated in Figure 2. In these cases, identity could be maintained during tracking by providing a more complex motion model that approximates the interaction between targets. We show below that incorporating an MRF to model interactions is equivalent to adding an

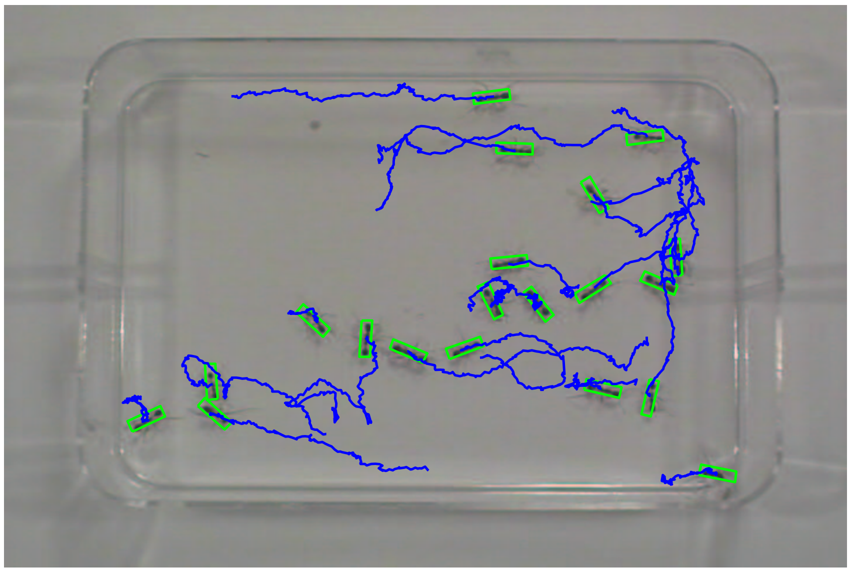
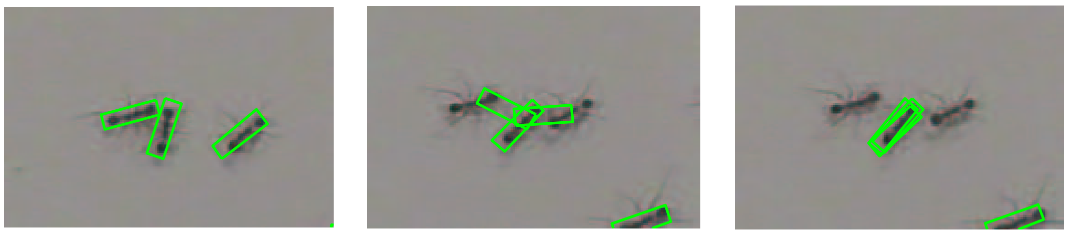


Figure 1: 20 ants are being tracked by an MCMC-based particle filter. Targets do *not* behave independently: whenever one ant encounters another, some amount of interaction takes place, and the behavior of a given ant before and after an interaction can be quite different. This observation is generally applicable to any situation where many interacting targets need to be tracked over time.



(a) frame 9043 (b) frame 9080 (c) 9083

Figure 2: (a) Three interacting ants are being tracked using independent particle filters. (b) The target with the best likelihood score typically "hijacks" the filters of nearby targets. (c) Resulting tracker failure. We address this problem using an Markov random field motion prior, built on the fly at each time step, that can adequately model these interactions and defeat these failure modes.

additional *interaction factor* to the importance weights in a joint particle filter.

The second contribution is to show how this can be done efficiently using Markov chain Monte Carlo (MCMC) sampling. The joint particle filter suffers from exponential complexity in the number of tracked targets, n. Computational requirements render the joint filter unusable for more than than three or four targets [20]. As a solution, we replace the traditional importance sampling step in the particle filter with an MCMC sampling step. This approach has the appealing property that *the filter behaves as a set of individual particle filters when the targets are not interacting, but efficiently deals with complicated interactions when targets approach each other*. The idea of using MCMC in the sequential importance resampling (SIR) particle filter scheme has been explored before, in [5]. Our approach can be consider a specialization of this work with an MRF-based joint posterior and an efficient proposal step to achieve reasonable performance.

In other related work, MCMC has been used in different ways in a particle filter setting. [13, 11, 1] introduce periodic MCMC steps to diversify particles in a fixed-lag smoothing scheme. Similarly, Marthi et. al. [23] developed "Decayed MCMC" sequential Monte Carlo, in which they focus the sampling activity of the MCMC sampler to state variables in the recent past. [29] uses a variational approximation rather than a sampling-based approximation scheme, and is of interest to see whether this could also be applied to multi-target target tracking.

Finally, several other particle-filter based approaches exist to tracking multiple identical targets. [28] "binds" particles to specific targets. [22] uses partitioned sampling and a probabilistic exclusion principle, which adds a term to the measurement model that assures that every feature measured belongs to only one target. BraMBLe [19] addresses tracking and initializing multiple targets in a variable-dimension framework. However, all of these are joint particle filter approaches and are less suitable to tracking a large number of targets.

2 Bayesian Multi-Target Tracking

The multiple target tracking problem can be expressed as a Bayes filter. We recursively update the posterior distribution $P(X_t|Z^t)$ over the joint state of the *all* n targets $\{X_{it}|i \in 1..n\}$ given all observations $Z^t = \{Z_1..Z_t\}$ up to and including time t, according to:

$$P(X_t|Z^t) = kP(Z_t|X_t) \int_{X_{t-1}} P(X_t|X_{t-1})P(X_{t-1}|Z^{t-1}) \tag{1}$$

The likelihood $P(Z_t|X_t)$ expresses the *measurement model*, the probability we observed the measurement Z_t given the state X_t at time t. The *motion model* $P(X_t|X_{t-1})$ predicts the state $X_t$at time t given the previous state X_{t-1}. In all that follows we will assume that the likelihood $P(Z_t|X_t)$ factors as across targets as $P(Z_t|X_t) = \prod_{i=1}^{n} P(Z_{it}|X_{it})$ and that the appearances of targets are conditionally independent.

2.1 Independent Particle Filters

When identical targets do *not* interact, we can approximate the exact Bayes filter by running multiple single-target *particle filters*. Mathematically, this is equivalent to factoring the motion model $P(X_t|X_{t-1})$ as $\prod_i P(X_{it}|X_{i,t-1})$.

For each of the n independent filters, we need to approximate the posterior $P(X_{it}|Z^t)$ over each target's state X_{it}. A particle filter can be viewed as an importance sampler for this posterior $P(X_{it}|Z^t)$, using the predictive density on the state X_{it} as the proposal distribution. Briefly, one inductively assumes that the posterior at the previous time step is approximated by a set of weighted particles

$$P(X_{it}|Z^{t-1}) \approx \{X^{(r)}_{,it-1}, \pi^{(r)}_{i,t-1}\}_{r=1}^{N}$$

Then, for the current time-step, we draw N samples $X^{(s)}_{it}$ from a proposal distribution

$$X^{(s)}_{it} \sim q(X_{it}) = \sum_r \pi^{(r)}_{i,t-1} P(X_{it}|X^{(r)}_{i,t-1})$$

which is a mixture of motion models $P(X_{it}|X^{(r)}_{i,t-1})$. Then we weight each sample so obtained by its likelihood given the measurement Z_{it}, i.e.

$$\pi^{(s)}_{i,t} = P(Z_{it}|X^{(s)}_{it})$$

This results in a weighted particle approximation $\{X^{(s)}_{it}, \pi^{(s)}_{it}\}_{s=1}^{N}$ for the posterior $P(X_{it}|Z^t)$ over the target's state X_{it} at time t. There are other ways to explain the particle filter (see e.g. [2]) that more easily accommodate other variants, but the mixture proposal view above is particularly suited for our application.

2.2 Independent Filter Failure Modes

While using independent filters is computationally tractable, the result is prone to frequent failures. Each particle filter samples in a small space, and the resulting "joint" filter's complexity is linear in the number of targets, n. However, in cases where targets *do* interact, as in an insect tracking scenario, single particle filters are susceptible to failures exactly when interactions occur. In a typical failure mode, illustrated in Figure 2, several trackers will start tracking the single target with the highest likelihood score.

3 MRF Motion Model

Our approach to addressing tracker failures resulting from interactions is to introduce a more capable motion model, based on Markov random fields (MRFs). We model the interaction between targets using a graph-based MRF constructed on the fly for each individual time-step. Formally, an MRF is a graph (V, E) with undirected edges between nodes where the joint probability is factored as a product of local potential functions at each node, and

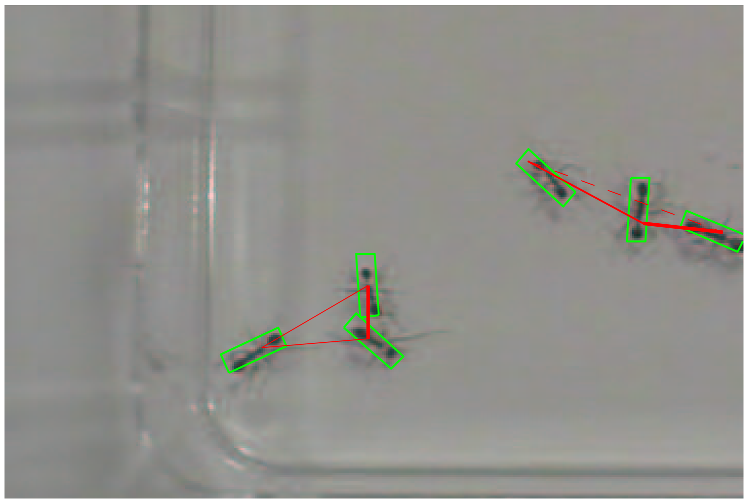


Figure 3: To model interactions, we dynamically construct a Markov random field at each time step, with edges for targets that are close to one another. An example is shown here for 6 ants. Targets that are far from one another are not linked by an edge, reflecting that there is no interaction.

interactions are defined on neighborhood cliques. See [30, 21] for a thorough exposition. The most commonly used form is a pairwise MRF, where the cliques are pairs of nodes that are connected in the undirected graph. We assume the following pairwise MRF form, where the $\psi(X_{it}, X_{jt})$ are pairwise interaction potentials:

$$P(X_t|X_{t-1}) \propto \prod_i P(X_{it}|X_{i(t-1)}) \prod_{ij \in E} \psi(X_{it}, X_{jt}) \tag{2}$$

The interaction potentials of the MRF afford us the possibility of easily specifying domain knowledge governing the joint behavior of interacting targets. At the same time, the absence of an edge in the MRF encodes the domain knowledge that targets do not influence each other's behavior. As a concrete example, in the insect tracking application we present in the Section 6, we know that two insects rarely occupy the same space. Taking advantage of this assumption can help greatly in tracking two targets that pass close to one another. An example MRF for our test domain is illustrated in Figure 3; in this case, targets within 64 pixels (about 2 cm) of one another are linked by MRF edges. The absence of edges between two ants provides mathematical rigor to the intuition that ants far away will not influence each other's motion.

Since it is easier to specify the interaction potential in the log domain, we express $\psi(X_{it}, X_{jt})$ by means of the Gibbs distribution:

$$\psi(X_{it}, X_{jt}) \propto \exp\left(-g(X_{it}, X_{jt})\right) \tag{3}$$

where $g(X_{it}, X_{jt})$ is a penalty function. For example, in the ant tracking application the penalty function $g(X_{it}, X_{jt})$ we use depends only on the number of pixels overlap between the target boxes of two targets. It is maximal when two targets coincide and gradually falls off as targets move apart.

4 The Joint MRF Particle Filter

The MRF terms that model interactions can be incorporated into the Bayes filter in a straightforward manner, but now we are forced to consider the full *joint* state of all n targets. In particular, analogous to the single target filter explained in Section 2.1, we recursively approximate the posterior on the joint state X_t as a set of N weighted samples, obtaining the following Monte Carlo approximation to the Bayes filter (1):

$$P(X_t|Z^t) \approx kP(Z_t|X_t) \sum_r \pi_{t-1}^{(r)} P(X_t|X_{t-1}^{(r)}) \tag{4}$$

We can easily plug in the MRF motion model (2) into the joint particle filter equation (4). Note that the interaction potential (3) does not depend on the previous target state X_{t-1}, and hence the target distribution (4) for the joint MRF filter factors as

$$P(X_t|Z^t) \approx kP(Z_t|X_t) \prod_{ij \in E} \psi(X_{it}, X_{jt}) \sum_r \pi_{t-1}^{(r)} \prod_i P(X_{it}|X_{i(t-1)}^{(r)}) \tag{5}$$

In other words, the interaction term moves out of the mixture distribution. This means that *we can simply treat the interaction term as an additional factor in the importance weight*. In other words, we sample from the joint proposal distribution function

$$X_t^{(s)} \sim q(X_t) = \sum_r \pi_{t-1}^{(r)} \prod_i P(X_{it}|X_{i(t-1)}^{(r)}) \tag{6}$$

and weight the samples according to the following factored likelihood expression:

$$\pi_t^{(s)} = \prod_{i=1}^{n} P(Z_{it}|X_{it}^{(s)}) \prod_{ij \in E} \psi(X_{it}^{(s)}, X_{jt}^{(s)})$$

4.1 Problems with the Joint MRF Filter

However, the joint particle filter approximation is not well suited for multi-target tracking. Each particle contains the joint position of *all* n targets, $X_t^{(s)} = \{X_{1t}^{(s)}, ..., X_{nt}^{(s)}\}$, and the filter suffers from exponential complexity in the number of tracked targets, n. If too few particles are used, all but a few importance weights will be near-zero. In other words, the Monte Carlo approximation (4), while asymptotically unbiased, will have high variance. These considerations render the joint filter unusable in practice for more than than three or four targets [].

5 The MCMC-based MRF Particle Filter

The second contribution of this paper is to show how that we can efficiently sample from the factored target posterior distribution (5) using Markov chain Monte Carlo (MCMC) sampling [24, 14, 10]. In effect, we are replacing the inefficient importance sampling step with an efficient MCMC sampling step.

All MCMC methods work by generating a sequence of *states*, in our case joint target configurations X_t at time t, with the property that the collection of generated states approximates a sample from the target distribution (5). To accomplish this, a Markov chain is defined over the space of configurations X_t such that the stationary distribution of the chain is exactly the target distribution.

The Metropolis-Hastings (MH) algorithm [17] is a way to simulate from such a chain. We use it to generate a sequence of samples from $P(X_t|Z^t)$. The pseudo-code for the MH algorithm is as follows (adapted from [14]) :

1. Start with a valid initial configuration X_t, then iterate once for each desired sample:
2. Propose a new assignment $X_t^{'}$ using the *proposal density* $Q(X_t'; X_t)$
3. Calculate the *acceptance ratio*
$$a = \frac{P(X_t'|Z^t)}{P(X_t|Z^t)} \frac{Q(X_t; X_t')}{Q(X_t'; X_t)} \tag{7}$$
4. If $a \geq 1$ then accept X_t' and set $X_t \leftarrow X_t'$. Otherwise, we accept it with probability a. If rejected we keep the state unchanged (i.e. return X_t as a sample).

5.1 Proposal Density 1

The key to the efficiency of this sampler rests in the specific proposal density we use. In particular, *we only change the state of one target at a time* by sampling directly from the factored motion model of the selected target

$$Q(X_t'|X_t) = \frac{1}{N} Q(X_t'|X_t, i) = \frac{1}{N} \sum_r P(X_{it}'|X_{i(t-1)}^{(r)}) \prod_{j \neq i} \delta(X_{jt}' = X_i)$$

Each target is equally likely to be selected. The acceptance ratio for this proposal can be calculated very efficiently, as only the likelihood and MRF interaction potential for the chosen target need to be evaluated:

$$a_{\mathbf{S}} = \min\left(1, \frac{P(Z_t|X_{it}') \prod_{j \in E_i} \psi(X_{it}', X_{jt}')}{P(Z_t|X_{it}) \prod_{j \in E_i} \psi(X_{it}, X_{jt})}\right)$$

This also has the desirable consequence that, if targets do *not* interact, the MCMC-based filter above is just as efficient as multiple, independent particle filters.

5.2 Proposal Density 2

One possible improvement to the proposal density from Section 5.1 is to select targets more frequently when they interact. On the assumption that the tracking is more complex for interacting targets, it might pay to devote more samples to those targets. In the results section, Section 4, we evaluate a second proposal density that does just that. While space limitations prevent us from giving the complete detail here, the algorithm is only slightly more complicated, but has two additional parameters that govern how much more samples are devoted to targets engaging in an interaction event.

5.3 Algorithm Summary

In summary, the detailed steps of the MCMC-based tracking algorithm we propose are:

1. At time $t-1$ the state of the targets is represented by an set of unweighted samples $\{X_{t-1}^{(r)}\}_{r=1}^{N}$. Each sample contains the joint state $X_{t-1}^{(r)} = \{X_{1(t-1)}^{(r)}, ..., X_{n(t-1)}^{(r)}\}$.
2. Initialize the MCMC sampler for time t by drawing X_t from the interaction-free predictive density $\sum_r \prod_i P(X_{it}|X_{i(t-1)}^{(r)})$, i.e. randomly select a joint sample $X_{t-1}^{(r)}$ and move all targets i in it according to the motion model $P(X_{it}|X_{i(t-1)}^{(r)})$.
3. Perform Metropolis-Hastings iterations to obtain M samples from the factored posterior (5). Discard the first B samples to account for sampler burn-in. In detail:
   (a) Proposal step:
      i. Randomly select a joint sample $X_{t-1}^{(r)}$ from the set of unweighted samples from the previous time step.
      ii. Randomly select a target i from n targets. This will be the target that we propose to move.
      iii. Using the previous state of this i^{th} target $X_{i(t-1)}^{(r)}$, sample from the conditionally dependent motion model $P(X'_{it}|X_{i(t-1)}^{(r)})$ to obtain X'_{it}
   (b) Compute the acceptance ratio:
   $$a_{\mathbf{S}} = \min\left(1, \frac{P(Z_{it}|X'_{it})\prod_{j\in E_i}\psi(X'_{it}, X'_{jt})}{P(Z_{it}|X_{it})\prod_{j\in E_i}\psi(X_{it}, X_{jt})}\right)$$
   (c) If $a_{\mathbf{S}} \geq 1$ then accept X'_{it} , set the the ith target in X_t to X'_{it}. Otherwise, we accept it with probability $a_{\mathbf{S}}$. If rejected, we leave the ith target in X_t unchanged.
   (d) Add a copy of the current X_t to the new sample set. (Note, if we are the burn-in phase, the MCMC sampler has not converged to the stationary distribution, we do not add X_t to the sample set.)
4. The sample set $\{X_t^{(s)}\}_{s=1}^{M}$ at time t represents an estimated joint state of the targets.

6 Experimental Validation

We evaluated our approach by tracking through a very long video-sequence of roaming ants (*Aphaenogaster cockerelli*), and present both quantitative results as well as a graphical comparison of the different tracker methodologies. The test sequence consists of 10,400 frames of 20 ants, roaming about an arena measuring 15 cm. by 10 cm. Frame rate was 30 Hz, and images are 720 by 480 24-bit RGB pixels. The ants themselves are about 1 cm. long and move about the arena as quickly as 3 cm per second. Interactions occur frequently and can involve 5 or more ants in close proximity. In these cases, the motion of the animals is difficult to predict. After pausing and touching one another, they often walk rapidly sideways or even backward. This experimental domain provides a substantial challenge to any multi-target tracker.

6.1 Experimental Details and Results

We evaluated a number of different trackers with respect to a baseline "pseudo ground truth" sequence. As no ground truth was available we obtained the baseline sequence by running a slow but accurate tracker and correcting any mistakes it made by hand. In particular, we ran our MCMC tracker with 2000 samples, which lost track only 15 times in the entire sequence. When we observed a tracker failure, we reinitialized by hand the positions of the targets and resumed tracking.

Below are the specific implementation choices we made to specialize the general algorithm of Section 5.3 to the ant tracking application:

- The *state* X_{it} of the i^{th} ant i is its position (x_{it}, y_{it}) in pixels and its orientation θ_{it}.
- For the *likelihood model* we used an appearance template approach with a robust error norm. In particular, we use a 10 by 32 pixel rectangular template containing a mean appearance image μ_F and a standard deviation image σ_F, both estimated from 149 manually selected ant images. We also learned a background mean image μ_B and standard deviation image σ_B from 10,000 randomly selected pixels. The log-likelihood is then calculated as

$$\log P(X_{it}|Z_t) = -\frac{1}{2}\frac{|F(X_{it}) - \mu_F|}{4\sigma_F} + \frac{1}{2}\frac{|F(X_{it}) - \mu_B|}{4\sigma_B}$$

  Here $F(X_{it})$ is the vector of pixels from a target with state X_{it} after translation and rotation to the template coordinate frame.
- For the *motion model* we used a normal density centered on the previous pose X_{t-1}

$$X_t|X_{t-1} = R(\theta_{t-1} + \Delta\theta)\begin{bmatrix} \Delta x & \Delta y & 0 \end{bmatrix}^\top + X_{t-1}$$

  where $[\Delta x, \Delta y, \Delta\theta] \sim [N(0, \sigma_x^2), N(0, \sigma_y^2), N(0, \sigma_\theta^2)]$ with $(\sigma_x, \sigma_y, \sigma_\theta) = (3.0, 5.0, 0.4)$.
- For the *MRF interaction terms* we used a simple linear interaction function γp where p is the area of overlap between two targets and $\gamma = 5000$.

Table 1: Tracker failures observed in the 10,400 frame test sequence

Tracker	**Proposal Density**	**Number of Samples**	**Number of Failures**
MCMC	1	50	123
MCMC	1	100	49
MCMC	1	200	28
MCMC	1	1000	16
MCMC	2	50	124
MCMC	2	100	42
MCMC	2	200	27
MCMC	2	1000	16
single particle filter	n/a	10 per target	148
single particle filter	n/a	50 per target	125
single particle filter	n/a	100 per target	119
joint particle filter	n/a	50	544
joint particle filter	n/a	100	519
joint particle filter	n/a	200	479
joint particle filter	n/a	1000	392

- *MCMC parameters*: we discard 25% of the samples to let the sampler burn in, regardless of the total number of samples.

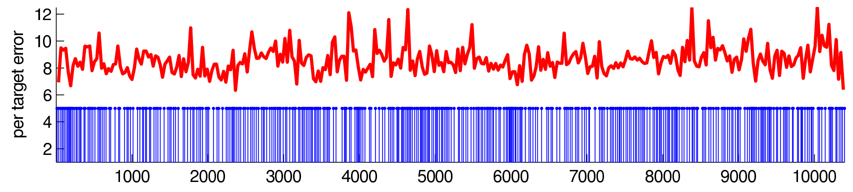
Table 1 shows the number of tracking failures for all the tracker/sample size combinations we evaluated. We automatically identified failures of these trackers when the reported position of a target deviated 50 pixels from the pseudo ground truth position. This allowed us to detect switched and lost targets without manual intervention.

Figure 4 shows the result graphically, comparing 4 different samplers, each with an equivalent sample size of 1000. For each of the trackers, we show exactly where failures occur throughout the sequences by tick-marks. At the bottom, in panel (e), we also show the number of interactions detected in each frame, which directly corresponds to the number of edges in the dynamical MRF motion model for that frame. To obtain a measure of trajectory quality, we also recorded for each frame the average distance of the targets to their ground truth trajectories. This is shown in the figure as a time series, for each tracker, averaged per second time unit.

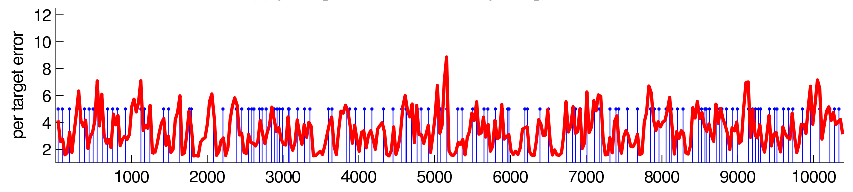
6.2 Discussion

From the quantitative results in Table 1 and the qualitative comparison in Figure 4 we draw the following conclusions:

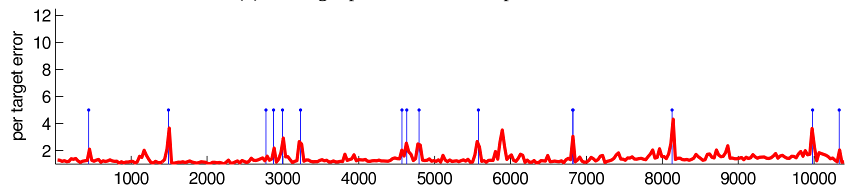
1. The joint filter is clearly unusable for tracking this many targets. The track quality is very low and number of errors reported is very high.



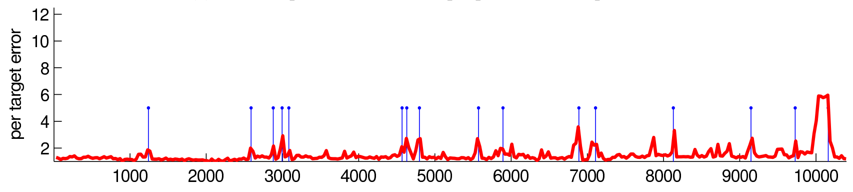
(a) joint particle filter, 1000 joint particles



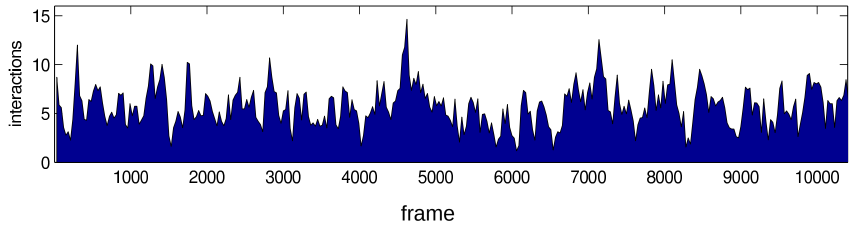
(b) 20 single particle filters, 50 particles each



(c) MCMC particle filter with proposal 1, 1000 particles

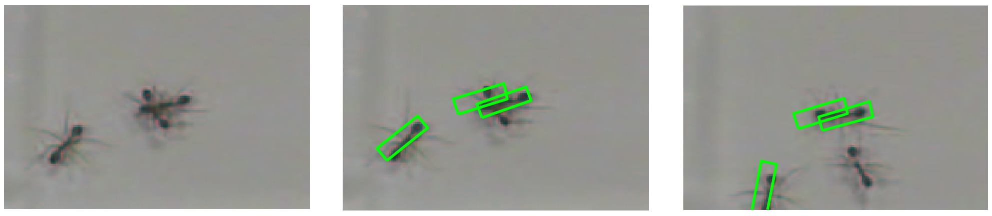


(d) MCMC particle filter with proposal 2, 1000 particles



(e) observed interactions

Figure 4: (a-d): Qualitative comparison of 4 trackers, each tracking 20 ants using an equivalent sample size of 1000. Tick marks show when tracking failures occur throughout the sequence. The time series plot shows average distance from ground truth (averaged per target and per second) (e) The number of automatically detected interactions between ants for each frame.



(a) frame 3054 (b) frame 3054 (c) frame 3072

Figure 5: Typical failure modes of MCMC-based MRF particle filter occur when the assumption that targets do not overlap is violated. (a) Two targets undergoing extensive overlap. (b) The tracker reports the incorrect position for the overlapped ant. (c) The resulting tracker failure.

2. The MCMC-based trackers perform significantly better than independent particle filters with a comparable number of samples, both in track quality and failures reported. For example, both MCMC trackers with 1000 samples had only 16 failures, as compared to 125 for 20 independent particle filters with 50 particles each.

3. To our surprise, an MCMC-based tracker with only 50 samples *total* performed as well as or better than 20 independent particle filters with 50 samples *each* (1000 samples total).

4. The MCMC-based trackers rapidly improve their performance as we increase the number of samples. The number of failures falls from 123 to 16 as the number of samples is increased from 50 to 1000. Such an effect is not seen for an equivalent increase in computation for the single particle filters, because in that case increasing the number of samples does not improve the ability to deal with ant interactions.

5. As expected, there is some correlation between the density of interactions, shown in Figure 4 (e), and the reported tracker failures. For example, the interaction peak near frame 4900 is likely to have contributed to the MCMC tracker failures that occur during and shortly after this peak.

6. In an interesting negative result, we found that the “smarter” proposal 2 did not in any way outperform proposal 1, at least not in the ant application.

7 Conclusions

In conclusion, the MCMC-MRF approach proposed in the paper has significantly improved the tracking of multiple interacting targets. Figure 5 shows that for the insect tracking case, the few remaining tracking failures that remain for the MCMC-based tracker occur when our assumption that targets do not overlap is violated. In these cases, it is unclear that any data-association method offers a solution. A more complicated joint likelihood model might be helpful in these cases.

In future work, we intend to validate the approach proposed here by tracking hundreds of interacting targets. Our long term research goals involve the analysis of multi-agent system behavior in general, with social insects as a model [3]. In particular, we are looking at honey bees in an active bee hive as a challenging test for multi-target tracking. On the algorithmic front, it would be interesting to more closely look at proposal densities that devote more computational power to interactions, despite the fact that we have not seen any significant performance improvement in our experiments so far. We are also looking into reversible jump MCMC [16] to automatically deal with a varying number of targets.

Finally, it is our hope that the MRF-based motion model and its efficient sequential implementation using MCMC will benefit other application domains besides vision-based animal tracking, for which it has clearly been shown to be useful.

Acknowledgments

This work was funded under NSF Award IIS-0219850.

References

[1] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan. An introduction to MCMC for machine learning. *Machine learning*, 50:5–43, 2003.

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

[3] T. Balch, Z. Khan, and M. Veloso. Automatically tracking and analyzing the behavior of live insect colonies. In *Proc. Autonomous Agents 2001*, Montreal, 2001.

[4] Y. Bar-Shalom, T.E. Fortmann, and M. Scheffe. Joint probabilistic data association for multiple targets in clutter. In *Proc. Conf. on Information Sciences and Systems*, 1980.

[5] C. Berzuini, N. G. Best, W.R. Gilks, and C. Larizza. Dynamic conditional independence models and Markov chain Monte Carlo methods. *Journal of the American Statistical Association*, 92:1403–1412, 1996.

[6] J. Carpenter, P. Clifford, and P. Fernhead. An improved particle filter for non-linear problems. Technical report, Department of Statistics, University of Oxford, 1997.

[7] I.J. Cox and J.J. Leonard. Modeling a dynamic environment using a Bayesian multiple hypothesis approach. *Artificial Intelligence*, 66(2):311–344, April 1994.

[8] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte Carlo Localization for mobile robots. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 1999.

[9] R. Deriche and O.D. Faugeras. Tracking line segments. *Image and Vision Computing*, 8:261–270, 1990.

[10] A. Doucet, N. de Freitas, and N. Gordon, editors. *Sequential Monte Carlo Methods In Practice*. Springer-Verlag, New York, 2001.

[11] A. Doucet, N. J. Gordon, and V. Krishnamurthy. Particle filters for state estimation of jump Markov linear systems. *IEEE Transactions on Signal Processing*, 49(3), 2001.

[12] T.E. Fortmann, Y. Bar-Shalom, and M. Scheffe. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE Journal of Oceanic Engineering*, 8, July 1983.

[13] W. Gilks and C. Berzuini. Following a moving target–Bayesian inference for dynamic Bayesian models. *Journal of the Royal Statistical Society, Series B*, 63(1):127–146, 2001.

[14] W.R. Gilks, S. Richardson, and D.J. Spiegelhalter, editors. *Markov chain Monte Carlo in practice*. Chapman and Hall, 1996.

[15] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Procedings F*, 140(2):107–113, 1993.

[16] P. Green. Reversible jump Markov chain Monte Carlo computation and bayesian model determination. *Biometrika*, 82:711–732, 1995.

[17] W.K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109, 1970.

[18] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *Eur. Conf. on Computer Vision (ECCV)*, pages 343–356, 1996.

[19] M. Isard and J. MacCormick. BraMBLe: A Bayesian multiple-blob tracker. In *Intl. Conf. on Computer Vision (ICCV)*, pages 34–41, 2001.

[20] Z. Khan, T. Balch, and F. Dellaert. Automatically tracking and analyzing the behavior of live insect colonies. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Las Vegas, 2003.

[21] S.Z. Li. *Markov Random Field Modeling in Computer Vision*. Springer, 1995.

[22] J. MacCormick and A. Blake. A probabilistic exclusion principle for tracking multiple objects. In *Intl. Conf. on Computer Vision (ICCV)*, pages 572–578, 1999.

[23] B. Marthi, H. Pasula, S. Russel, and Y. Peres. Decayed MCMC filtering. In *Proceedings of the 18th Annual Conference on Uncertainty in AI (UAI)*, 2002.

[24] R.M. Neal. Probabilistic inference using Markov chain Monte Carlo methods. Technical Report CRG-TR-93-1, Dept. of Computer Science, University of Toronto, 1993.

[25] C. Rasmussen and G.D. Hager. Probabilistic data association methods for tracking complex visual objects. *PAMI*, 23(6):560–576, June 2001.

[26] D.B. Reid. An algorithm for tracking multiple targets. *IEEE Trans. on Automation and Control*, AC-24(6):84–90, December 1979.

[27] D. Schulz, W. Burgard, D. Fox, and A. B. Cremers. Tracking multiple moving targets with a mobile robot using particle filters and statistical data association. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2001.

[28] D. Tweed and A. Calway. Tracking many objects using subordinate Condensation. In *British Machine Vision Conference (BMVC)*, 2002.

[29] J. Vermaak, N. Lawrence, and P. Perez. Variational inference for visual tracking. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 773–780, 2003.

[30] G. Winkler. *Image analysis, random fields and dynamic Monte Carlo methods*. Springer Verlag, 1995.