Efficient Particle Filter-Based Tracking of Multiple Interacting Targets Using an MRF-based Motion Model

Zia Khan, Tucker Balch, and Frank Dellaert
{zkhan,tucker,dellaert}@cc.gatech.edu
College of Computing, Georgia Institute of Technology
Atlanta, Georgia 30332, USA

***Abstract*— We describe a multiple hypothesis particle filter for tracking targets that will be influenced by the proximity and/or behavior of other targets. Our contribution is to show how a Markov random field motion prior, built on the fly at each time step, can model these interactions to enable more accurate tracking. We present results for a social insect tracking application, where we model the domain knowledge that two targets cannot occupy the same space, and targets will actively avoid collisions. We show that using this model improves track quality and efficiency. Unfortunately, the joint particle tracker we propose suffers from exponential complexity in the number of tracked targets. An approximation to the joint filter, however, consisting of multiple nearly independent particle filters can provide similar track quality at substantially lower computational cost.**

I. 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.

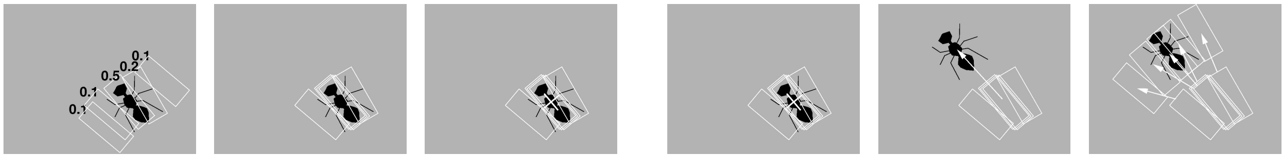
For non-interacting targets, the classical multi-target tracking literature approaches the problem by performing a data-association step after a detection step. Most notably, the multiple hypothesis tracker [1] and the joint probabilistic data association filter (JPDAF) [2], [3] are the most influential 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 [4], the multiple hypothesis tracker in [5], and the JPDAF in [6]. Recently, a particle filtering version of the JPDAF has been proposed in [7].

These approaches are appropriate when targets behave independently, and the problem is one of visual confusion. However, in some applications, e.g. when tracking social agents, the targets do *not* behave independently. For example, in our own work on tracking social insects like ants and bees [8], the insects effectively live on a 2D plane and will rarely crawl on top of each other. In these cases, identity could be maintained during tracking by providing a more complex motion model that models the interaction between targets.

Our contribution is to show how a Markov random field motion prior, built on the fly at each time step, can model these interactions. Our approach is based on the well known particle filter [9], [10], [11], [12], a multi-hypothesis tracker that approximates the filtered posterior distribution by a set of weighted particles. The standard particle filter weights particles based on a likelihood score, and then propagates these weighted particles according to a motion model. We show that incorporating an MRF to model interactions is equivalent to adding an additional *interaction factor* to the importance weights in a joint particle filter.

Unfortunately the joint particle filter suffers from exponential complexity in the number of tracked targets, n. However, we have found that using n *nearly* independent particle filters, each running in a low-dimensional space, can approximate the optimal joint tracker by computing the interaction potentials based on single-target predictions from the previous time step. The result is a set of trackers that are coupled only when necessary. The performance advantage the decoupled trackers in comparison with a joint filter is significant. Computational requirements render the joint filter unusable for more than than three or four targets, while our decoupled tracking system can easily track 20 or more targets.

We have selected visual animal tracking as a domain to illustrate the approach. In particular, we track a number of ants in a small arena. This is not an artificial task: our long term research goals involve the analysis of multi-agent system behavior, with social insects as a model. The domain offers many challenges that are quite different from the typical radar tracking domain in which most multi-target tracking algorithms are evaluated.



(a) (b)

Fig. 1. (a) Particle filtering: a set of particles (white rectangles), are scored according to how well the underlying pixels match the appearance model (left). Particles are resampled (middle) according to the normalized weights determined in the previous step. Finally, the estimated location of the target is computed as the mean of the resampled particles. (b) Motion model: The previous image and particles (left). A new image frame is loaded (center). Each particle is advanced according to a stochastic motion model (right). The samples are now ready to be scored and resampled as above.

II. THE PARTICLE FILTER

Particle filters use multiple discrete "particles" to represent the belief distribution over the location of a tracked target [9], [10], [11], [12]. This part of our system is not new, but we review the algorithm to introduce notation and to facilitate understanding of the new components.

In the Bayes filtering paradigm, we recursively update the posterior distribution over the current state X_t given all observations $Z^t = \{Z_1..Z_t\}$ up to and including time t, as follows:

$$\begin{aligned} P(X_t|Z^t) &= kP(Z_t|X_t)P(X_t|Z^{t-1}) \\ &= kP(Z_t|X_t)\int_{X_{t-1}} P(X_t|X_{t-1})P(X_{t-1}|Z^{t-1}) \end{aligned}$$

where the likelihood $P(Z_t|X_t)$ expresses the *measurement model* and $P(X_t|X_{t-1})$ is the *motion model*. In a particle filter we approximate the posterior $P(X_{t-1}|Z^{t-1})$ recursively as a set of weighted N samples $\{X_{t-1}^{(r)}, \pi_{t-1}^{(r)}\}_{r=1}^N$, where $\pi_{t-1}^{(r)}$ is the weight for particle $X_{t-1}^{(r)}$. Given this, we can use a a Monte Carlo approximation of the integral and get:

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

One way to view a particle filter is as an importance sampler for this distribution. Specifically, N samples $X_t^{(s)}$ are drawn from the *proposal distribution*

$$q(X_t) = \sum_r \pi_{t-1}^{(r)} P(X_t|X_{t-1}^{(r)})$$

and then weighted by the likelihood, i.e.

$$\pi_t^{(s)} = P(Z_t|X_t^{(s)})$$

This results in a weighted particle approximation $\{X_t^{(s)}, \pi_t^{(s)}\}_{s=1}^N$ for the posterior $P(X_t|Z^t)$ at time t. Note that there are other ways to explain the particle filter (see e.g. [13]) that also accommodate other variants, but the mixture proposal view above is particularly suited for our application.

The general operation of the particle filter tracker is illustrated in Figure 1 for tracking ants in video. In our case, each hypothesis (or particle) is represented by a rectangular region about the size of our ant targets. Each target is tracked by 5 particles in the illustrated example, but in actual runs we typically use 200 particles per target. We assume we start with particles distributed around the target to be tracked. The principal steps in the algorithm include:

1) **Score:** each particle is scored according to how well the underlying pixels match an appearance model, by weighting it with $P(Z_t|X_t)$ (see next section).
2) **Resample:** the particles are "resampled" according to their score. This operation results in the same number of particles, but very likely particles are duplicated while unlikely ones are dropped. Note that this is equivalent to choosing a mixture component to sample from in (1).
3) **Average:** the location of the target is estimated by computing the mean of all the associated particles. This is the estimate reported by the algorithm as the location of the target in the current video frame.
4) **Apply motion model:** each particle is stochastically repositioned according to a model of the target's motion. This is implemented by sampling from the chosen mixture component in (1).

III. APPEARANCE-BASED TRACKING

For completeness, we describe here the details of the appearance based measurement model we used for tracking insects in video, given that all the results we show in Section V are based on this. We assume that the measurement Z_t consists of a local image neighborhood I, formed as the superposition of a sprite with shape s on a background (this is similar to [14], but without occlusion). In particular, for the ant-tracking experiments below we model the sprite as a rectangular image with three parameters: length, width, and position of rotational center (along the length).

In the following, we refer to the appearance model parameters as θ_f. We also model the background of the image with parameters θ_b. An estimate for the background model is obtained by averaging all the images over the entire video sequence. For notational convenience, let $x = X_t$ in what follows. Assuming that pixel values in I are conditionally independent given x, we can factor

$P(I|x,\theta)$ into foreground and background components:

$$P(I|x,\theta) = \prod_{p \in B(s,x)} P(I_p|\theta_b) \prod_{p \in F(s,x)} P(I_p|x,\theta_f)$$

where p is an individual pixel, and

- $\prod_{p \in B(s,x)} P(I_p|\theta_b)$ is the likelihood factor due to the background pixels $B(s,x)$. The set of background pixels $B(s,x)$ depends on sprite shape s and position x.
- $\prod_{p \in F(s,x)} P(I_p|x,\theta_f)$ is the likelihood factor due to pixels part of the foreground. The set of foreground pixels $F(s,x)$ again depends on sprite shape s and position.

Because the sets $B(s,x)$ and $F(s,x)$ add up to I, we can simplify this as

$$P(I|x,\theta) = k' \prod_{p \in F(s,x)} \frac{P(I_p|x,\theta_f)}{P(I_p|\theta_b)} \tag{2}$$

where the new constant k' does not depend on the sprite shape s or target state x. This is a large computational win, as (2) is simply the product of the likelihood ratios in the foreground area $F(s,x)$.

The exposition above is independent of the actual pixel measurement models used. Below we use a simple appearance model for both background and foreground model, modeling pixels as corrupted versions of a known foreground and background model. Using zero-mean Gaussian additive noise, the log-likelihood $\log P(I|x,\theta)$ is then equal to:

$$C + \frac{1}{2} \sum_{p \in F} \left(\frac{I_p - \mu_{bp}}{\sigma_{bp}} \right)^2 - \left(\frac{I_p - \mu_f(p,x)}{\sigma_{fp}} \right)^2 \tag{3}$$

where C is a constant. We use Equation 3 as our score for the appearance underlying a particular sample. Note that the terms in the background sum can be pre-computed once for the entire image and stored. The second sum of squares is simply a transformation of the template, and a subtraction from the image. Multi-channel images are easily accommodated, by noting that p can index both location and channel (e.g., for RGB images).

IV. MULTI-TARGET TRACKING

This section describes the main contribution of the paper. We show how multiple targets can be modeled using a joint particle tracker, using a Markov random field (MRF) model for how they interact. We then show how to approximate this correct but intractable joint filter with multiple nearly independent trackers.

A. The Joint Particle Filter

A joint particle filter is quite similar to a simple particle filter. The primary difference is that each "particle" estimates the locations of all the targets being tracked. Thus in Equation 1 X_t refers to the **joint state of all the targets**. An important consequence of the joint representation is that each particle is d^n-dimensional, where d is the dimensionality of an individual filter. As a consequence, if N particles are necessary for reliable tracking of a single target, N^n are typically required for tracking n target. Tracking multiple targets with joint hypotheses is an exponentially complex problem both in terms of space and computational time.

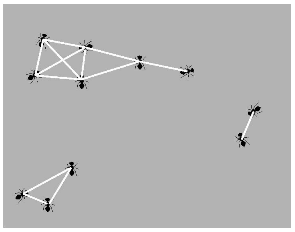


Fig. 2. Example Markov random field for a group of ants. Only targets that are close to one another have the potential for interacting in their motion. Links in the graph indicate this potential. Targets that are very far from one another do not have an edge connecting them – reflecting that there is no interaction.

In what follows, we first discuss the joint appearance-based likelihood model, and then show how we can modify the basic particle filter equation 1 above to incorporate interactions between different targets.

B. Joint Appearance Model

First, we assume that the joint appearance likelihood $P(Z_t|X_t)$ factors over the different targets X_{it}, with $i \in 1..n$:

$$P(Z_t|X_t) = \prod_{i=1}^{n} P(Z_{it}|X_{it})$$

This assumes targets will not occlude each other. This assumption is in fact not needed for the joint tracker, but is crucial in approximating the joint tracker with n independent trackers as in Section IV-E.

C. Markov Random Field Interaction and Blocking

The benefit of using a joint tracker is that we can model non-trivial interactions between targets at the motion model stage, i.e. intelligent targets that take account of each other's position in order to avoid collisions. If the targets do not interact at all, the joint motion model $P(X_t|X_{t-1})$ also factors over all targets as $\prod_j P(X_{jt}|X_{j(t-1)})$, and one could just use n independent particle filters.

We model the interaction between targets by a Markov random field (MRF) constructed on the fly for the current time-slice. 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 interactions are defined on neighborhood cliques. See

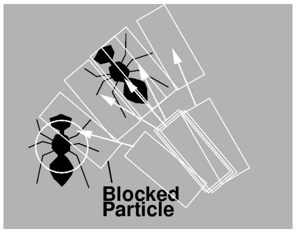


Fig. 3. Blocking. Samples relating to one target are penalized if they fall within a certain radius of the estimated location of another target.

[15], [16] 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 will assume the following pairwise MRF form:

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

where the $\psi(X_{it}, X_{jt})$ are pairwise *interaction potentials*.

The particular interaction we model in this paper is the domain knowledge in the insect tracking application that two insects will not 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 2. In our experiments the MRF constructed at each time step links targets within 30 pixels (about 1 cm) of one another. 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:

$$\psi(X_{it}, X_{jt}) \propto \exp\left(-g(X_{it}, X_{jt})\right) \tag{4}$$

where $g(X_{it}, X_{jt})$ is a penalty function. In our application, we penalize joint particles where some targets overlap the location of another target. Figure 3 illustrates this idea. The penalty function $g(X_{it}, X_{jt})$ we used only depends on the distance $d(X_{it}, X_{jt})$ between two targets, is maximal when two targets coincide, and gradually falls off as targets move apart. This models the fact that targets will not overlap, as is appropriate in the insect tracking application.

D. The Joint MRF Particle Filter

Note that the interaction potential (4) does not depend on the previous target state X_{t-1}, and hence the target posterior distribution (1) for the joint MRF filter factors as

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

i.e. 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

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

by drawing a mixture component at random, and then moving each individual target independently. Then, we weight each of the particles $X_t^{(s)}$ so obtained by

$$\pi_t^{(s)} = \prod_{i=1}^{n} P(Z_{it}|X_{it}^{(s)}) \prod_{ij\in E} \psi(X_{it}^{(s)}, X_{jt}^{(s)}) \tag{5}$$

E. Split Trackers

Importance sampling is notoriously inefficient in high-dimensional state spaces: if not enough particles are used, all but a few particles will have a near zero-weight. As a result, the Monte Carlo approximation for the posterior, while asymptotically unbiased, will have high variance. We illustrate this fact in the results section below.

We hypothesize that n *nearly* independent trackers, where n is the number of targets, may provide trajectories nearly as well as the optimal joint tracker, while requiring substantially less computation. The trackers are not fully independent, because they consider the locations of other targets when scoring particles. In particular, we penalize particles that overlap the locations of other targets (as described above for the joint tracker).

The scoring in the independent trackers is a simplification of the scoring in the joint tracker, as follows:

$$\pi_{it}^{(s)} = P(Z_{it}|X_{it}^{(s)}) \prod_{ij\in E} \psi(X_{it}^{(s)}, \bar{X}_{j(t-1)})$$

where $\bar{X}_{j(t-1)}$ is the estimated state for target j at time $t-1$, i.e. the mean of the particles of the j^{th} tracker.

V. EXPERIMENTAL RESULTS

Multi-target trackers are commonly evaluated using real or simulated ground-based radar data of airborne targets [2]. Radar data is typified by a sparse distribution of small targets. Target motion is easy to predict because the targets are subject to momentum and acceleration constraints. In contrast, our work is demonstrated on video of 20 1.0 cm long ants (*Aphaenogaster cockerelli*) roaming about an arena measuring 15 cm by 10 cm (6 inches by 4 inches). The ants move about the arena as quickly as 3 cm per second and they encounter one another frequently. The motion of these animals is difficult to predict – they often walk sideways or even backward. This experimental domain provides a substantial challenge.

The test data for these experiments was gathered by videotaping the ants' activities for 30 minutes. We selected a 30 second interval at random to analyze. Image frames are collected at 30 Hz, thus the test sequence includes of 900 frames. Each frame is composed of 720 by 480 pixels of 24 bits coded as RGB intensities. We applied

tracker	tracks	correct tracks	percent correct	total particles	run time
		no block / block	no block / block		(seconds)
joint	1	1 / 1	100 / 100	200	54 / 54
joint	2	1 / 2	50 / 100	40,000	8,113 / 8,240
split	1	1 / 1	100 / 100	200	53 / 54
split	2	1 / 2	50 / 100	400	87 / 115
split	10	8/ 9	80 / 90	2,000	451 / 464
split	20	15 / 17	75 / 85	4,000	711 / 716

TABLE I

QUANTITATIVE PERFORMANCE OF THE JOINT AND SPLIT TRACKERS WITH AND WITHOUT BLOCKING.

all variations of our algorithm to this same 900 frame sequence. The algorithm is coded in the CML language and compiled and run on a 2.5 GHz Pentium-4 machine under Linux. We make use of Intel's IPP library for some image processing operations.

Results were evaluated on the following criteria

1) **Run time:** the total time to process 900 frames in seconds.
2) **Correct tracks:** the total number of targets tracked from start to end of the sequence without errors. Correctness is evaluated manually by comparing the computed trajectories against the actual paths of the ants.

Numerical results are summarized in Table I.

A. Results with a Joint Tracker without Blocking

The prohibitive run time of the joint tracker limited the number test cases we could examine. We tested the joint tracker by having it track one and two ants with 200, and 200^2 particles respectively. Run time for one ant was 54 seconds to process the entire 30-second video segment, and run time for two ants was 8113 seconds (2 hours, 15 minutes). The particular ants we chose to track interacted in such a way that the track for one ant was lost and transferred to the other. Figure 4 illustrates the interaction and the track failure.

B. Results with a Joint Tracker and Blocking

We tested MRF Blocking in the joint filter over the same test case. The tracker successfully tracks both ants through the interaction that confused the tracker without blocking. The sequence is illustrated in Figure 5. Run times for the two cases were the similar to those without blocking.

C. Results with Split Trackers

With independent or "split" trackers, run time performance is significantly improved, and a more comprehensive evaluation is possible. We evaluated runs tracking 1, 2, 10 and 20 ants with 200, 400, 2000 and 4000 total particles respectively. Run times were 53 seconds, 87 seconds, 451 seconds and 711 seconds.

Without blocking enabled, the split trackers frequently fail to correctly trace a target's trajectory. Many errors occur, similar to the one illustrated in Figure 4. In the case of 1 ant, the tracker does not lose the ant, but for 10 tracks there are 2 failures and for 20 tracks there are 5 failures.

D. Results with Split Trackers and MRF Blocking

We evaluated runs tracking 1, 2, 10 and 20 ants with 200, 400, 2000 and 4000 total particles respectively. Run times were 54 seconds, 115 seconds, 464 seconds and 716 seconds.

In all three multi-track cases (2, 10 and 20 ants) with blocking enabled, split trackers are more likely to trace a target's trajectory correctly. We also note that the two ants examined in Figure 4, for which the joint tracker without blocking failed, are tracked successfully by split trackers with blocking. The performance improvements are summarized in Table I.

VI. CONCLUSIONS

We examined the problem of tracking multiple interacting targets with an MRF-augmented particle filter. While a theoretically optimal joint filter is too computationally expensive to track more than two targets at once, we show that multiple nearly independent trackers (with an interaction term) generate tracks of similar quality to those generated by a joint tracker, but at a substantially lower cost.

The blocking prior from the insect-domain is but one example of what can potentially be modeled in an MRF framework. In that domain, it is the most salient interaction that can be observed without assuming more about the particular social role of each insect. Mutual attraction and short stereotyped social behaviors could potentially be modeled using an MRF, as well.

However, more complex interactions are possible when conditioning on insect roles. For exampe, we are looking at the elaborate dance of honey bees as an example of behavior that probably go beyond the capabilities of an MRF, and will necessitate more complex machinery.

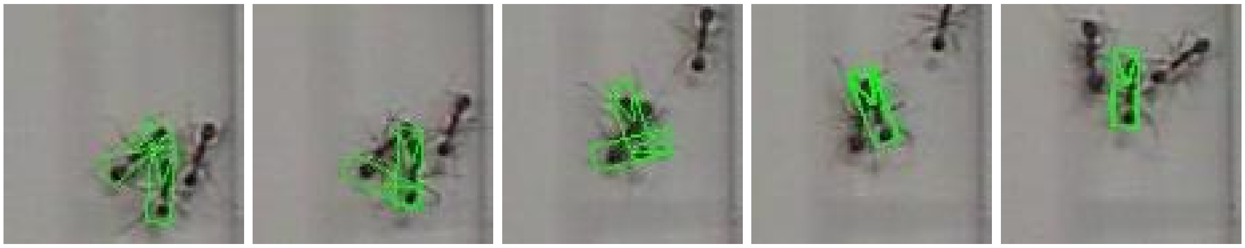


Fig. 4. Two ants interact and cause the track of one of them to fail.

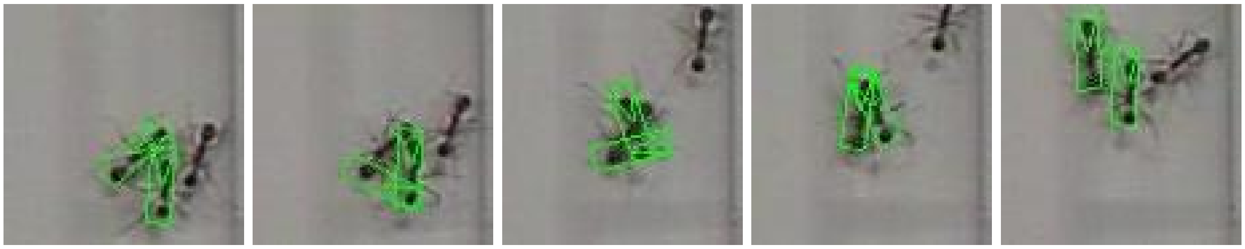


Fig. 5. The same interaction is tracked successfully with MRF blocking.

VII. ACKNOWLEDGMENTS

This work was funded under NSF Award IIS-0219850.

VIII. REFERENCES

[1] D. Reid, “An algorithm for tracking multiple targets,” *IEEE Trans. on Automation and Control*, vol. AC-24, pp. 84–90, December 1979.

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

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

[4] R. Deriche and O. Faugeras, “Tracking line segments,” *Image and Vision Computing*, vol. 8, pp. 261–270, 1990.

[5] I. Cox and J. Leonard, “Modeling a dynamic environment using a Bayesian multiple hypothesis approach,” *Artificial Intelligence*, vol. 66, pp. 311–344, April 1994.

[6] C. Rasmussen and G. Hager, “Probabilistic data association methods for tracking complex visual objects,” *PAMI*, vol. 23, pp. 560–576, June 2001.

[7] 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 Int. Conf. on Robotics and Automation (ICRA)*, 2001.

[8] T. Balch, Z. Khan, and M. Veloso, “Automatically tracking and analyzing the behavior of live insect colonies,” in *Proc. Autonomous Agents 2001*, (Montreal), 2001.

[9] N. Gordon, D. Salmond, and A. Smith, “Novel approach to nonlinear/non-Gaussian Bayesian state estimation,” *IEE Procedings F*, vol. 140, no. 2, pp. 107–113, 1993.

[10] M. Isard and A. Blake, “Contour tracking by stochastic propagation of conditional density,” in *Eur. Conf. on Computer Vision (ECCV)*, pp. 343–356, 1996.

[11] J. Carpenter, P. Clifford, and P. Fernhead, “An improved particle filter for non-linear problems,” tech. rep., Department of Statistics, University of Oxford, 1997.

[12] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, “Monte Carlo Localization for mobile robots,” in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 1999.

[13] 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*, vol. 50, pp. 174–188, Feb. 2002.

[14] N. Jojic and B. Frey, “Learning flexible sprites in video layers,” in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2001.

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

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