Efficient Object Tracking in Video Sequences by means of LS-N-IPS

Péter Torma
Eötvös Loránd University,
Mindmaker Ltd.
e-mail: tyus@mindmaker.hu

Csaba Szepesvári
Mindmaker Ltd.
H-1121 Budapest,
Konkoly Thege Miklós út 29-33.

Abstract

A recently introduced particle filtering method, called LS-N-IPS, is considered for tracking objects on video sequences. LS-N-IPS is a computationally efficient particle filter that performs better than the standard N-IPS particle filter, when observations are highly peaky, as it is the case of visual object tracking problems with good observation models. An implementation of LS-N-IPS that uses B-spline based contour models is proposed and is shown to perform very well as compared to similar state-of-the art tracking algorithms.

1 Introduction

The incremental change between consecutive image frames is always a great help when considering vision based object tracking problems. There are two main approaches in the field that exploit this redundancy. One of them uses the assumption that the object does not move much from frame to frame and employs a local search around the previous object position to locate the object on the next frame (see e.g. [4]). Algorithms in this group tend to be very precise when locked on the object but may have problems if the environment is highly cluttered, or unexpectedly big motion occurs between the frames. The other approach makes use of some filtering algorithm (e.g. [1, 7, 8]), most notably particle filtering methods(see e.g [9, 5, 12]. The particle filters keep multiple hypothesis about the object state and this increases its robustness against clutter. However, particle filters often give very crude position information unless an excessively large number of particles is used.

N-IPS is a successful particle filter method[1], also known as CONDENSATION in the image processing literature [7]. N-IPS, however, suffers from inefficiency problems if the observation density is uninformative and/or uniformly very small except in a small neighborhood of the true state since then particles which are not in this small vicinity of the "true" state will all have roughly equal observation likelihoods and the filter becomes effectively decoupled from the observations. In order to simplify the exposition, we shall call such densities "peaky" throughout this article.

[1]The name N-IPS is the abbriviation of N-Interacting Particle System used by [11]

In this article we consider a recent modification of N-IPS, called LS-N-IPS [14] in visual tracking problems. Since LS-N-IPS was designed to overcome the problem of N-IPS with peaky observation densities, therefore it is natural to consider it in visual tracking problems. LS-N-IPS combines local search with particle filtering and thus can be thought of as a combination of the two main streams of vision based tracking research mentioned above. As a consequence, the algorithm inherits the high precision of local search based object tracking methods and the robustness of the particle filter based methods, even when a small number of particles is used. In this article a specific implementation that uses object contours is described in detail and results of a real-world experiment are discussed.

The article is organized as follows: In Section 2 we define the filtering problem and discuss the problems of N-IPS with peaky densities in some more detail. In Section 3 LS-N-IPS is presented and some insight is provided on its behavior. In Section 4 a possible implementation of LS-N-IPS for the object tracking is described in detail. Details of experiments and results on a real-world object tracking problem are presented in Section 6. A short discussion of related work is presented in Section 7, whilst conclusions are drawn and future work is outlined in Section 8.

2 The filtering problem

Let us consider the filtering problem defined by the system

$$X_{t+1} = f(X_t) + W_t, \quad (1)$$
$$Y_t = g(X_t) + V_t, \quad (2)$$

where $t = 0, 1, 2, \ldots$, $X_t, W_t \in \mathcal{X}$, $Y_t, V_t \in \mathcal{Y}$, and W_t, V_t are zero mean i.i.d. random variables. X_t is the state and

Y_t is the observation at time t. Let the posterior given the observations $Y_{0:t} = (Y_0, \ldots, Y_t)$ be π_t:

$$\pi_t(A) = P(X_t \in A | Y_{0:t}),$$

where $A \subset \mathcal{X}$ is any measurable set. The filtering task consists of inferring the posterior given the past observations. For convenience, we shall also use the symbols f and g to denote the respective densities $f(X_{t+1}|X_t)$ and $g(Y_t|X_t)$.

Particle filtering works by maintaining a finite number of hypothesises of the states $(X_t^{(1)}, \ldots, X_t^{(N)})$, sometimes called the particles. The system of particles is aimed to represent the posterior π_t by means of the empirical posterior $\hat{\pi}_t$ given by $\hat{\pi}_t(A) = (1/N)\sum_{i=1}^N \chi_A(X_t^{(i)})$. Here χ_A denotes the characteristic function of A.

N-IPS works by repeatedly updating the positions of the particles individually by Equation 1 and then resample them using the distribution that is proportional to $(g(Y_t|X_t^{(1)}), \ldots, g(Y_t|X_t^{(N)}))$.

Assume that the observation density values are uninformatively very small except when $X_t^{(i)}$ is very close to X_t, i.e. $g(Y_t|X_t^{(i)}) \approx \epsilon$ independently of Y_t whenever $d(X_t, X_t^{(i)}) \geq \rho$. Here d is some appropriate metric and $\rho > 0$ is some small number.

Now, if the number of particles is not sufficiently high then with high probability $d(X_t, X_t^{(i)}) \geq \rho$ will hold for most of the particles, especially if the dynamics of the system is noisy (with $\mathrm{Var}(W_t) >> \rho$). In such a case the particle system will quickly loose the object and will start to perform a random walk as the observations will not have any influance on the states of the particles. The LS-N-IPS filter to be presented in the next section was designed to overcome this problem.

3 The LS-N-IPS algorithm and its properties

The algorithm is as follows:

1. **Initialization:** Let $X_0^{(i)} \sim \pi_0$, $i = 1, 2, \ldots, N$ and set $t = 0$.
2. Repeat forever:
   (a) **Prediction:** Compute the proposed next states by $Z_{t+1}^{(i)} = S_\lambda(f(X_t^{(i)}) + W_t^{(i)}, Y_{t+1})$, $i = 1, 2, \ldots, N$.
   (b) **Evaluation:** Compute $w_{t+1}^{(i)} \propto g(Y_{t+1}|Z_{t+1}^{(i)})$, $i = 1, 2, \ldots, N$.
   (c) **Resampling:**
      i. Sample $k_{t+1}^{(i)} \sim (w_{t+1}^{(1)}, \ldots, w_{t+1}^{(N)})$, $i = 1, 2, \ldots, N$.
      ii. Let $X_{t+1}^{(i)} = Z_{t+1}^{(k_{t+1}^{(i)})}$, $i = 1, 2, \ldots, N$.
   (d) t:=t+1

As it should be clear, the only difference between LS-N-IPS and N-IPS is the way they update the positions of the particles. LS-N-IPS uses a non-trivial local search operator, S_λ, to "refine" the "crude" predictions $f(X_t^{(i)}) + W_t^{(i)}$, whilst N-IPS uses the trivial operator $S_\lambda(x, y) = x$.

One should think of S_λ as a local search for refining a particle state such that the observation likelihood given the current observation Y_{t+1} is maximized. The purpose of this is to make the particles become more "relevant".

3.1 Uniform Convergence of LS-N-IPS

Provided that the system to be filtered is sufficiently regular, the N-IPS model is known to provide a uniformly "good" estimate of the posterior, for any given particle-set size (see Theorem 3.1 of [11]).

Our aim in this section is to state an analogous result for LS-N-IPS. Since LS-N-IPS uses the search operator S_λ instead of the true dynamics, in general, we cannot hope that it would improve on the bounds derived for N-IPS. The theorem, to be stated below, is still important as it shows the stability of LS-N-IPS, i.e., that the tracking error can be kept bounded and reduced (to a limit) by increasing the number of particles. In the next subsection we will argue that LS-N-IPS can indeed improve on the performance on N-IPS under the special conditions discussed earlier.

Since LS-N-IPS can be viewed as an N-IPS algorithm where in each step an approximate dynamics is used in the prediction step, we give the theorem for the general case when approximate models are used in N-IPS. (This case is interesting on its own right since the models are usually learnt from data and hence are approximate themselves).

First, we need some definitions. Let K be the Markov transition kernel corresponding to Equation 1, π_0 be the prior over the states. The approximate models we consider will be represented by their Markov transition kernels $\hat{K}_t$ (i.e., the approximation may depend on time), the approximate prior $\hat{\pi}_0$, and a (stationary) approximate observation model $\hat{g}$.

Let h be the Hilbert projective metrics defined by $h(f, g) = \ln \sup_{x,x'} f(x)g(x')/(f(x')g(x))$, where $x, x' \in \mathcal{X}$ and $f, g \in L^1(\mathcal{X})$ (cf. [6]). Let $C(K)$ be defined by

$$C(K) = \ln \sup_{x,y,x',y'} \frac{K(x,y)K(x',y')}{K(x',y)K(x,y')}.$$

Further, let $G_y : L^1(\mathcal{X}) \to L^1(\mathcal{X})$ be defined by $(G_y f)(x) = g(y|x)f(x)$ and let $F : L^1(\mathcal{X}) \to L^1(\mathcal{X})$ be the Markov operator corresponding to K: $(Ff)(x) = \int K(x,y)f(y)dy$. Let $\hat{G}_y$ and $\hat{F}_t$ be defined analogously.

Theorem 3.1. *Assume that for some $\hat{\varepsilon} > 0$ and $\hat{a} > 0$*

$$\frac{1}{\hat{a}} \leq \hat{g}(y|x), g(y|x) \leq \hat{a} \tag{3}$$

and

$$\hat{\varepsilon} \leq \hat{K}_t(x,z), K(x,z) \leq \frac{1}{\hat{\varepsilon}}. \tag{4}$$

Further, assume that for some $\beta > 0$ these approximate entities satisfy

$$\sup_y h(G_y, \hat{G}_y), h(F, \hat{F}_t), h(\pi_0, \hat{\pi}_0) \leq \beta.$$

Consider the N-IPS model based on the approximate models and the approximate prior. Consider the empirical posterior $\hat{\pi}_t^N$ as computed by this N-IPS algorithm. Then the uniform bound

$$\sup_{t\geq 0} E(\,|\int f d\hat{\pi}_t^N - \int f d\pi_t|\,|\,Y_{0:t}\,) \leq \frac{5\exp(2\hat{\gamma}')}{N^{\hat{\alpha}/2}} + \frac{4\beta}{\log(3)(1-\tanh(C(K)/4)} \tag{5}$$

holds for any $N \geq 1$ and for any measurable function f satisfying $\|f\|_\infty \leq 1$, where $\hat{\gamma}'$ and $\hat{\alpha}$ are defined by

$$\hat{\alpha} = \frac{\hat{\varepsilon}^2}{\hat{\varepsilon}^2 + \hat{\gamma}'} \quad \text{with} \quad \hat{\gamma}' = 1 + 2\log\hat{a}.$$

The proof that uses the properties of h and the fundamental stability theorem of [11] can be found in [14].

Let $\hat{g} = g$, $\hat{\pi}_0 = \pi_0$ and $\hat{K}_t$ to be the kernel corresponding to the dynamics induced by

$$S_\lambda(f(X_t) + W_t, Y_t) =$$

$$p(f(X_t) + W_t) + (1-p)c\chi_{[\hat{X}_t - \delta/2, \hat{X}_t + \delta/2]}(X_t)$$

where c is a normalisation constant making $\int f(.) + W_t = \int c\chi_{[\hat{X}_t-\delta/2,\hat{X}_t+\delta/2]}$, $\hat{X}_t = \operatorname{argmax}\{g(Y_t|\tilde{x})\,|\,\|\tilde{x} - f(X_t) + W_t\| \leq \lambda\,\}$ and δ is some small noise parameter. One can think this S_λ as identity operator with probability p, and a noisy local search operator with probability $1-p$. Provided that the original dynamics K_t satisfy the conditions (3) and (4), the above theorem can be interpreted as providing conditions on S_λ under which stability of tracking is maintained along with some bounds on the tracking error. For this note that

$$h(F_t, \hat{F}_t) \leq \ln \frac{\frac{1}{\hat{\epsilon}}\frac{\lambda}{\hat{\epsilon}\delta}}{\hat{\epsilon}\frac{\hat{\epsilon}}{p}} = \ln\frac{\lambda p}{\hat{\epsilon}^4\delta}.$$

4 Object tracking using B-spline contour models

In this section we propose an implementation of LS-N-IPS for object tracking using B-spline contour models.

In order to implement LS-N-IPS one needs to define three objects: the dynamics used in the prediction step, the local search operator and the observation density. In our case the state space will consist of the pose of the object to be tracked along with the previous pose (the dynamics is a second order AR process). The pose defines a contour mapped onto the camera plane (i.e., onto the image).

The idea is to use this B-spline based contour model as the basis of the observation calculations: the better the contour fits the image, the more likely the corresponding pose is. B-splines, however will be represented here by their support points (i.e. by points along the curve) as opposed to the usual represenation of B-splines via their control points.

This is needed since the local search is implemented on the image by finding an allowable spline contour in the vicinity of the original one that fits the image the best in that given small neighbourhood. This search is implemented by finding the most likely locations of edges along the normals of the original curve *at the support points*. By using support points instead of control points we spare some matrix vector products that would involve matrices whose size scales with the complexity of the contour. On the other hand, the usage of support points lefts the complexity of the algorithm intact otherwise.

In the following subsections we provide the details of the algorithm.

4.1 Spline contours, configuration space

Let us consider the spline curve $s : [0, L] \to \mathcal{R}^2$ defined by its support points $q^x = (q_1^x, \ldots q_n^x)^T, q^y = (q_1^y, \ldots q_n^y)^T$, i.e: $s(t) = ((A^{-1}q^x)^T\varphi(t), (A^{-1}q^y)^T\varphi(t))$, where $\varphi(t) = (\varphi_1(t), \varphi_2(t), \ldots, \varphi_n(t))^T$ are the usual B-spline basis functions (cf. [2]) and $s(i) = (q_i^x, q_i^y)^T$. A is linear transformation mapping control points to support points and depends only on φ.

Let $q_0 = ((q_0^x)^T, (q_0^y)^T)$ be a vector of support points defining the contour s_0. If G is a group of similarity transformations of the 2D plane ($\mathcal{R}^2$) then one can find a matrix $W = W_{G,q_0}$ such that $T \in G$ iff for some $z \in \mathcal{R}^k$, the support vector $q = Wz + q_0$ yields the spline curve $T s_0$ (we use the convention that $z = 0$ corresponds to $T = \mathrm{Id}$). $\mathcal{R}^k$ then corresponds to the set of allowed configurations of the object.

Note 1. *If G is the group of Euclidean similarities of the plane then*

$$W = \begin{pmatrix} \mathbf{1} & \mathbf{0} & q_0^x & -q_0^y \\ \mathbf{0} & \mathbf{1} & q_0^y & q_0^x \end{pmatrix}, \tag{6}$$

where $\mathbf{0} = (0, 0, \ldots, 0)^T$, $\mathbf{1} = (1, 1, \ldots, 1)^T$.

4.2 Implementation of the Local Search

Assume that a contour (s) corresponding to some pose (z) is given. Blake and Isard define the likelihood of the contour given the image (the observation) as the product of the individual "likelihoods" of edges being located at some measurement points along the spline curve [7].

Motivated by this definition, our local search algorithm searches for maximal edge 'likelihood' values along the normals in the vicinity of the measurement points, the neighborhood itself defined by the search length, $l > 0$. The measurement points are chosen to be the support points $(q_1^x, q_1^y)^T, \ldots, (q_n^x, q_n^y)^T$.

Assume that the search yields the points $(\hat{q}_1^x, \hat{q}_1^y)^T, \ldots (\hat{q}_n^x, \hat{q}_n^y)^T$ and let $\hat{q} = (\hat{q}_1^x, \ldots \hat{q}_n^x, \hat{q}_1^y, \ldots \hat{q}_n^y)^T$ (for an example see Figure 1). Let $\hat{s}$ be the spline curve corresponding to $\hat{q}$.

The next step is to find a configuration whose corresponding spline curve matches $\hat{s}$ the best: $\hat{z} = \operatorname{argmin}_{z \in \mathcal{R}^k} \|s_z - \hat{s}\|_2^2$. Here s_z denotes the spline curve corresponding to the configuration z, i.e., the spline curve corresponding to the support vector $q_z = Wz + q_0$.

It is well known that if s is the spline contour corresponding to q then $||s||_2^2$ can be expressed as a function of q alone. In particular,

$$||s||_2^2 = \frac{1}{L}\int_0^L s^2(t)dt = q^T \begin{pmatrix} \mathcal{B} & 0 \\ 0 & \mathcal{B} \end{pmatrix} q = q^T \mathcal{U} q,$$

where

$$\mathcal{B} = A^{-T}\left(\frac{1}{L}\int_0^L \varphi(u)\varphi(u)^T du\right) A^{-1}.$$

Therefore, if we let $\|q\|_S^2 = q^T U q$ be the weighted ℓ^2 norm then $\hat{z} = \operatorname{argmin}_{z \in \mathcal{R}^k} \|Wz + q_0 - \hat{q}\|_S^2$. Now, by standard LMS calculations $\hat{z} = W^+(\hat{q} - q_0)$, where $W^+ = (W^T \mathcal{U} W)^{-1} W^T \mathcal{U}$. The corresponding projected support vector shall be denoted by $q^\perp = W\hat{z} + q_0$.

It remains to specify the likelihood of a given particle $\hat{z}$ (the previous configuration is not used in the observation likelihood calculations as still images reflect only momentarily information). The ideal likelihood would be the product of the likelihoods of the edges now measured at the points defined by the new support vector $q^\perp$. However, in favour of speed we would like to reuse the maximal likelihoods found during the search process. Therefore we start with the product of these and compensate for the errors introduced by this approximation.

Firstly, note that it is reasonable to devalve this value by a value proportional to the distortion caused by the projection, i.e. by $||q^\perp - \hat{q}||_S$: the larger the distortion is the less the likelihood of the given configuration should be, since a large distortion means that the spline curve $\hat{s}$ was far from $G\{s_0\}$.

However, $||q^\perp - \hat{q}||_2$ must be normed by the "scale" of the corresponding configurations: remember that $||q^\perp - \hat{q}||_2$ equals to the $\mathcal{L}^2$ distance between the spline curves corresponding to $q^\perp$ and $\hat{q}$, i.e., the area in between these curves. Now if $q^\perp$ and $\hat{q}$ correspond to "big" contours then $||q^\perp - \hat{q}||_2$ will be "big" itself. Therefore (provided that G includes the subgroup defined by scale transformations) $||q^\perp - \hat{q}||_2$ must be scaled by the size of the represented object. In a somewhat ad hoc way, we have chosen to norm this by d^2, where d is the scale of the object being in the configuration $z^\perp$. The search length was also scaled by d, so that the search length will correspond to a fixed search length of the physical world. In summary, the algorithm works as follows:

Evaluation Algorithm with Local Search. (Inputs: image (I), configuration (z))

1. Calculate the scaling factor d given z.
2. Calculate the search length $l = dl_0$, where l_0 is a user defined parameter.
3. For all normal of s at (q_i^x, q_i^y), where $q = Wz + q_0$ $(i = 1, 2, \ldots, n)$:
   - For every point on the normal and within the search length l calculate the likelihood of an edge being at that point.
   - Select the maximum value along the normal.
4. Compute v, the product of the obtained maximum values, and let the locations of the maximums define $\hat{q}$.
5. Calculate the best LMS configuration
$$z^\perp = W^+(\hat{q} - q_0)$$
and the projected support vector $q^\perp = Wz^\perp + q_0$.
6. Return
$$c = v\frac{d^2}{||q^\perp - \hat{q}||_2}$$

Figure 1 shows a contour corresponding to a single particle before (white) and after the local search (grey). It should be clear from the figure that the local search adjusted contour has a much higher likelihood of being the correct contour of the hand on the given image.

5 Experiments

The proposed algorithm was tried in a number of visual object tracking problems. In the scenarios shown here the hand-tracking was attempted. Edge likelihoods along the

Figure 1. *Predicted(white) and local-search adjusted(grey) contours.(see text)*

normals were computed as the product of the edge strength along the normal and the color match "inside"the object. For color matching a Gaussian density is used that is trained on the first frame. In the present implementation users have to draw the contour of the object to be tracked on the first frame by determining the support points of the curve. For G, the full Euclidean similarity group of the plane was chosen with W given by Equation 6, and thus the scale of an object with configuration $z = (z_1, z_2, z_3, z_4)^T$ is given by $d = \sqrt{(1 + z_3)^2 + z_4^2}$.

The state is defined by the pose and the previous pose (i.e., $\mathcal{X} = \mathcal{R}^8$). The pose space is defined as the parameterization of the transformation group defined by the subgroups induced by x, y-translations, rotations, and scaling, i.e., the pose space and the configuration space are non-linearly related. A second order dynamics was fitted for each dimension of the pose space, independently of each other. The dynamics of scale and rotation were adjusted by hand due to insufficient training data.

Note that conditions of the stability theorem holds if we set the noise on the dynamics and each contour likelihood bounded below with some positive bound. This is desired intuitively, as in case of occlusion all hypothesyses must have positive probability.

6 Results

The proposed algorithm was tested in a number of tracking tasks. In general, the algorithm performed very well under a wide range of conditions even with very small particle sizes, such as $N = 50$. Thanks to relying mostly on contour (i.e. shape) information, the algorithm could work under a wide range of illumination conditions, even when the lighting was weak.

Objects were tracked reliably and precisely (see Figures 1 and 2). In case of sudden movements the tracker occassionally lost the object but it could quickly recover in all of the cases. This can be explained by the relatively high variance of the noise of the dynamics and the high specificity of the observation likelihood: If the object is lost then all particles become roughly equally weighted and the system starts to perform a random diffusion, exploring the image. If some particle becomes in the vicinity of the true object position the local search refines the particle position quickly, the weight of the corresponding particle becomes large and the particle is reproduced with high probability in many copies in the resampling step. This causes the tracker to recover. If the shape information is not sufficiently specific then the tracker may lock on spurious "objects" having a contour similar to that of the object to be tracked. However, this is a common property of contour based tracking algorithms and is thus not special to LS-N-IPS.

Figure 2 shows every 15th frame of a typical tracking session. This image sequence was recorded at 30 frames/second. The number of particles was chosen to be 100, $l_0 = 10$. The number of contour control points was $n = 25$. The image resolution was 240×180.

Processing time measurements were made on a 1 GHz Pentium 4 computer. These show that the time spent on a single frame in terms of the control point number n can be well approximated by $0.031 + 0.0015(n - 25)sec$ (for $10 < n < 40$). It is independent of image resolution and scales linearly with the number of particles. The prediction and update steps require negligable time. Most of the time is spent on calculating the observation likelihoods and the local search procedure. We have also made measurements with the local search switched off (and using the unmodified observation likelihoods). This corresponds to running N-IPS (or CONDENSATION). We found that running the algorithm in this way, $N = 200$ particles requires roughly the same amount of processing time than that of needed by LS-N-IPS with $N = 100$. However, N-IPS was pretty much useless unless N was raised above 2000 which would allow a tracking speed of less than $5 frame/second$, to be compared with the real-time tracking speed $(30 frames/second)$ obtained with the proposed algorithm.

7 Related Work

Sequential importance sampling with resampling [10, 13, 5], or SIR (also known as iCONDENSATION [8] in the image processing community) is designed to overcome problems related to peaky *posteriors*. The design parameter of this algorithm is a so-called proposal distribution whose purpose is to concentrate particles to the highly probable parts of the state space.

Figure 2. *Tracking hand in clutter with* 100 *particles. Black contours show particles having high observation likelihoods, whilst the white contour shows the predicted contour of the hand.*

Unfortunatelly, good proposal distributions are not easy to design. The ideal proposal distribution depends both on the dynamics and the most recent observation. However, a proposal should also be fast to sample from. Blake and Isard suggest to use a density fitted to the output of a color detector as the proposal distribution. The for of the density in their application is a mixture of Gaussians, so it is cheap to sample from it. Unfortunately however, since the importance function is not defined on the speed of the object, motion coherence information is used by this algorithm in a limited way.

Another closely related work is a multiple hypothesis figure tracking [3] by T.Cham and J.Rheng. They use search procedure to update a mixture of gaussians posterior. Their algorithm is more closely related to an extended Kalman filter approach, though no theoretical analysis is mentioned. Howerver they found the same experimental results as this paper establishes, that multiple hypothesis filtering algorithms combined with local seach procedures performs more efficient on visual tracking tasks.

8 Conclusions and Future Work

An implementation of LS-N-IPS was proposed and examined in detail for visual object tracking problems. The algorithm was found to perform consistently better than competing algorithms both in terms of computation time, tracking precision and recovery time.

Future work will include implementing poligon trackers for higher dimensional spaces to track 3 dimensional rotations and translations. This, combined with a shading model, can be capable of tracking 3-dimensional movements of real objects. As the generative models of rotation and shading is well known in the computer graphics literature, with the help of simple local linearization algorithms LS-N-IPS is hoped to perform more efficient than other state-of-the-art figure tracking algorithms.

References

[1] R. C. Andrew Blake and A. Zisserman. A framework for spatio-temporal control in the tracking of visual contours. *Int. J. Computer Vision*, 1993.

[2] A. Blake and M. Isard. *Active Contours*. Springer-Verlag, 1998.

[3] T. Cham and J. Rheng. A multiple hypothesis approach to figure tracking. *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, II:239–245, 1999.

[4] G. D.Hager and P. N.Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. on PAMI*, to appear.

[5] A. Doucet. On sequential simulation based methods for Bayesian filtering. *Statistics and Computing*, 10(3):197–208, 1998.

[6] G.Birkhoff. *Lattice Theory*. Am. Math. Soc., 1967.

[7] M. Isard and A. Blake. CONDENSATION – conditional density propagation for visual tracking. *International Journal Computer Vision*, 29:5–28, 1998.

[8] M. Isard and A. Blake. ICondensation: Unifying low-level and high-level tracking in a stochastic framework. *Proc 5th European Conf. Computer Vision*, 1998.

[9] L. Jun and R. Chen. Sequential monte carlo methods for dynamic systems. *J. Amer. Statist. Assoc.*, pages 1032–1044, 1998. (to appear).

[10] T. M.A. and W. W.H. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 1987.

[11] P. D. Moral. A uniform convergence theorem for the numerical solving of the nonlinear filtering problem. *Journal of Applied Probability*, 35:873–884, 1998.

[12] P. D. Moral and S. G. Non-linear filtering using monte carlo particle methods. *C.R. Acad. Sci. Paris*, pages 1147–1152, 1995.

[13] D. Rubin. A noniterative sampling/importance resampling alternative to the data augmentation algorithm for creating a few imputations when fractions of missing information are modest: The sir algorithm, comment on tanner and wong(1987). *Journal of the American Statistical Association*, 1987.

[14] P. Torma and C. Szepesvári. LS-N-IPS: an improvement of particle filters by means of local search. *NOLCOS, St Petersburg, Russia Proceedings*, 2001. (to appear).