Pattern Recognition 42 (2009) 2372 – 2391

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Variance reduction techniques in particle-based visual contour tracking

Daniel Ponsa[a,*], Antonio M. López[b]

[a]Centre de Visió per Computador, Universitat Autònoma de Barcelona, Edifici O, Bellaterra 08193, Spain
[b]Centre de Visió per Computador & Dept. Ciències de la Computació, Universitat Autònoma de Barcelona, Edifici O, Bellaterra 08193, Spain

ARTICLE INFO

Article history:
Received 25 June 2008
Received in revised form 30 March 2009
Accepted 4 April 2009

Keywords:
Contour tracking
Active shape models
Kalman filter
Particle filter
Importance sampling
Unscented particle filter
Rao-Blackwellization
Partitioned sampling

ABSTRACT

This paper presents a comparative study of three different strategies to improve the performance of particle filters, in the context of visual contour tracking: the unscented particle filter, the Rao-Blackwellized particle filter, and the partitioned sampling technique. The tracking problem analyzed is the joint estimation of the global and local transformation of the outline of a given target, represented following the active shape model approach. The main contributions of the paper are the novel adaptations of the considered techniques on this generic problem, and the quantitative assessment of their performance in extensive experimental work done.

1. Introduction

Visual contour tracking is an area of research that has received much attention by the computer vision community for many years. One essential reason for this to happen is that, in many application domains, the contour of an object is a very informative cue about its state or configuration. Proof of that is the application of contour tracking in areas like visual surveillance [1], traffic monitoring [2], medical diagnosis [3,4] and human–machine interaction [5,6], among others.

The tracking of contours has been posed mainly as a minimization or as an inference problem. Following the first perspective, the so-called *active contour* methods adapt iteratively an elastic curve to image edges, while imposing some constraints on it (e.g., smoothness and compactness). The *classical* snakes approach [7] performs that by minimizing an energy term associated to a parametric curve. Geodesic active contours [9], which generalize in most situations classical snakes [10], pose the problem from a geometric point of view. Targets are segmented using an implicit contour representation. A non-parametric surface is evolved according to image edges, being the tracked contour the zero level set of this surface. The main advantage of this *level set-based* approach is that topological changes of the original curve are naturally managed. Extensions of this work, where contours are defined in terms of the content of the region that they enclose conform the *active regions* methods [11–13].

An important disadvantage of minimization-based approaches is the possibility of converging into local minima and mistrack the target. This drawback can be treated in a principled way by posing contour tracking as an inference problem. Now the goal is estimating the posterior density of a contour given image observations. Minimization-based approaches can be interpreted as a way of determining the maximum a posteriori of this density, assuming implicitly its unimodality. Problems appear when this density is not unimodal, which can be eluded if the whole density is estimated. This paper studies contour tracking from this perspective.

Formally, given a parametric model of the contour to be tracked, the goal is estimating at each instant t the probability density function (PDF) of the model parameters $\mathbf{x}_t$ (i.e., the contour state), conditioned on the observations up to t (i.e., $\mathbf{y}_{1:t} = [\mathbf{y}_i]_{i=1}^{t}$). In many applications this PDF can be properly assumed Gaussian, and its parameters can be efficiently estimated by means of Kalman-based filters. However, in cluttered scenes, this Gaussian assumption is usually too rough, since the PDF presents in fact multiple modes. This happens when there is more than one model parameterization that fits tightly to image observations, due to the presence of the tracked shape and also of other distractors in the scene. In these cases, it seems reasonable to maintain more than one contour tracking hypothesis, and in that way assure to keep track of the one that effectively adjusts to the object of interest. A principled manner to perform that consists in representing $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ by means of a population of *particles* (i.e., concrete

* Corresponding author. Tel.: +34 935812440; fax: +34 935811670.
E-mail addresses: daniel@cvc.uab.cat (D. Ponsa), antonio@cvc.uab.cat (A.M. López).

0031-3203/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2009.04.007

$\mathbf{x}_t$ instances), distributed (ideally) according to this PDF. In that way, any arbitrary form of the filtering density can be properly managed, what results in a tracking performance more robust to clutter. Providing a proper particle-based representation of $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is the objective pursued by the so-called particle filters (PFs). Briefly, PFs are stochastic sampling methods that sequentially approximate $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ by combining a particle-based representation of this density at the previous instant $t-1$ (i.e. $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$), and new collected observations $\mathbf{y}_t$. Due to that, they are commonly referred as sequential Monte Carlo methods and good reviews of their theoretical basis can be found in [14–16]. PFs were seminally applied to the problem of contour tracking by Isard and Blake [17,18], in a particular form that they termed as Condensation algorithm.

The Condensation algorithm is definitely the most popular form of PF applied in vision-based tracking applications. However, its computational cost (which depends on the amount of particles needed to represent $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ properly) increases exponentially with the number of parameters of the target model used. That is, it suffers from the curse of dimensionality. This is a serious drawback in contour tracking problems. In general, targets being tracked present global transformations of their outline (e.g., translations, rotations, etc.), as well as simultaneous local shape deformations. Consequently, the dimension of the parametric model of the contour is rather big, what makes the cost of its robust tracking high. The good news is that, since the problem of Condensation with the state dimensionality is well known, different generic strategies have been proposed to counteract it. In this paper we analyze the performance of three of these strategies in the context of visual contour tracking: the unscented particle filter (UPF), the Rao-Blackwellized particle filter (RBPF), and the partitioned sampling (PS) technique. We contribute with their adaptation in the context of contour tracking using active shape model (ASM). Developed mainly by British research groups in Leeds, Oxford and Manchester [19–21], ASMs represent the outline of an object by means of a parametric model, whose representability is limited to a given space of transformations, whether generic (e.g., Euclidean or affine transformations of a basic shape) or specific (shape deformations *learned* from the statistic analysis of training data). Our study focuses on ASMs since exploit naturally the a priori knowledge on the feasible shapes that a target can take. As traditionally formulated, they do not consider topological changes of the contour. However, in [22] is shown that applying their same principles on implicit contour representations, a parametric model is obtained that can manage such cases. ASMs have been shown effective in many application domains [20], and thanks to their parametric nature, their use in inference-based contour tracking is direct. However, as it is shown in [23,24], strategies exist to consider also non-parametric contour representations inside this framework.

As will be stated in the respective sections, two of the three techniques studied in this paper (the UPF and the PS) have also been applied previously in the contour tracking problem by other authors. However, our proposals differ significantly from the ones in these previous works. On the one hand, we use a more complete contour model, accounting for global and local shape transformations. On the other, our model of the contour observation process is more rigorous and accurate, leading to a better interpretation of the evidence extracted from frames. Another contribution of this paper is an exhaustive study of the performance of the proposed algorithms. This has been done using synthetic sequences, distorted with different levels of noise. Using the knowledge of the parameters used to generate the sequences, the performance of each technique has been measured quantitatively. This has allowed us to rank proposed algorithms at each evaluated situation, and to identify their strengths and weaknesses. Algorithms have also been tested on real sequences, in the contexts of hand and pedestrian racking.

Table 1
List of acronyms.

AR	Auto-regressive
AR1	Auto-regressive process of first order
ASM	Active shape model
CBM	Constrained Brownian motion
IPPF	Independent partition particle filter
KF	Kalman filter
MCE	Mean contour error
OISD	Optimal importance sampling density
PDF	Probability density function
PF	Particle filter
PF-EIS	Particle filtering with efficient importance sampling
PS	Partitioned sampling
RB	Rao-Blackwellization
RBPF	Rao-Blackwellized particle filter
SIS	Sequential importance sampling
SISR	Sequential importance sampling with resampling
SNR	Signal-to-noise ratio
UKF	Unscented Kalman filter
UPF	Unscented particle filter

The remainder of this paper is organized as follows: Section 2 gives an overview of the used contour model representation (the ASM), and introduces the approach used to jointly account for the affine transformations and the local deformations of a given shape of interest. Then, Section 3 focuses on modeling the shape evolution along time, and Section 4 on how the shape model relates to observations in images. Section 5 formalizes the visual tracking of contours as a Bayesian inference problem, and presents the general solution to this problem given by the importance sampling technique, which has led to the so-called particle filtering. The main drawbacks of this approach are remarked, and three different strategies to deal with them are adapted in the following sections to the contour tracking problem: the UPF (Section 6), the RBPF (Section 7), and the PS (Section 8). A comparative study of the performance of these approaches is presented in Section 9, and final conclusions are provided in Section 10. A list of the abbreviations used in the paper is given in Table 1.

2. Contour representation

In many application domains the use of shape tracking algorithms is motivated by the need not only to localize a given target, but also to identify its specific pose or configuration. To fulfill that, a generative model of the target shape variability is required. Many authors have worked on developing representations of shape variability in many different ways. Refs. [19–21] review the major contributions on this field, and then focus on the description of a model-based approach to shape tracking, commonly denoted as the ASM approach. There exist different possibilities to represent parametrically the outline of an object. In the ASM formalism, the dominant approach is based on 2D contours modeled by B-spline parametric curves.

B-splines construct expressions of a 2D contour as a weighted sum of N_B basis functions. Contour point coordinates $\mathbf{r}(s)=[x(s)\ y(s)]^T$ of 2D shapes are obtained by an expression of the form

$$\begin{bmatrix} x(s) \\ y(s) \end{bmatrix} = \begin{bmatrix} \mathbf{B}(s)^T & \mathbf{0}_{N_B}^T \\ \mathbf{0}_{N_B}^T & \mathbf{B}(s)^T \end{bmatrix} \begin{bmatrix} \mathbf{q}^x \\ \mathbf{q}^y \end{bmatrix},$$

or more compactly $\mathbf{r}(s)=\mathbf{U}(s)\mathbf{q}$. We denote with $\mathbf{0}_{N_B}$ a column vector of N_B zero elements. $\mathbf{B}(s)$ is a vector maintaining N_B basis functions, which are curves composed of polynomials of degree d with finite support. They are C^{d-1} continuous, which means that the contour derivatives up to the $(d-1)$-th are smooth. $\mathbf{q}=[\mathbf{q}^x\ \mathbf{q}^y]^T$ is the vector of the N_{cp} *control points* that weights the basis functions to generate a desired curve. Thus, its dimensionality corresponds to $N_q=2N_{cp}$, being $N_{cp}=N_B$. The parameter s evaluates the linear combination of

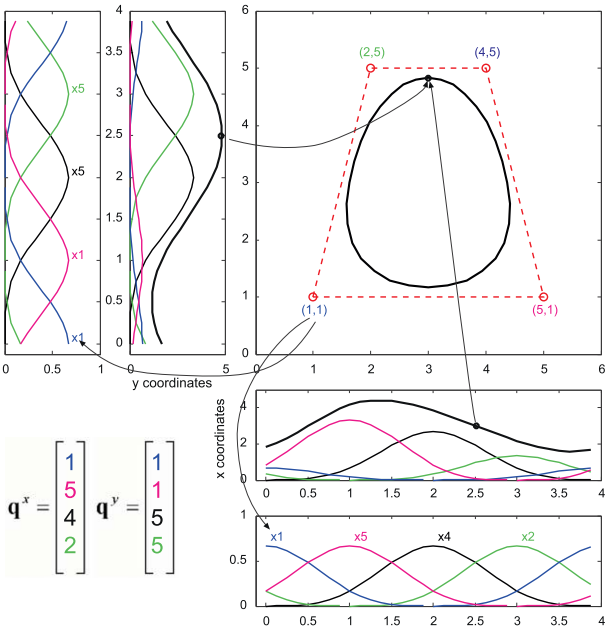

Fig. 1. Spline contours in a nutshell.

polynomial at concrete points. Fig. 1 sketches graphically the elements involved in the spline-based contour synthesis.

In practice, contours are not commonly synthesized from any arbitrary value of s, but at predefined discrete values. By fixing a sampling ratio N_{sr} between control points, a discrete contour is synthesized, consisting of N_s 2D coordinates (x,y), with $N_s = N_{cp}N_{sr}$. Curve sampling points s_i are regularly placed along the support of the spline parametric space, given by $\{s_i = i/N_{sr}\}_{i=0}^{N_s-1}$. Thus, a sampled contour representation is obtained with

$$\begin{bmatrix} x(s_0)\\ \vdots \\ x(s_{N_s-1})\\ y(s_0)\\ \vdots \\ y(s_{N_s-1})\end{bmatrix} = \begin{bmatrix} \mathbf{B}(s_0)^T & \mathbf{0}_{N_B}^T\\ \vdots & \vdots \\ \mathbf{B}(s_{N_s-1})^T & \mathbf{0}_{N_B}^T\\ \mathbf{0}_{N_B}^T & \mathbf{B}(s_0)^T\\ \vdots & \vdots \\ \mathbf{0}_{N_B}^T & \mathbf{B}(s_{N_s-1})^T\end{bmatrix} \begin{bmatrix}\mathbf{q}^x\\ \mathbf{q}^y\end{bmatrix},$$

or in compact form $\mathbf{r} = \mathbf{U}\mathbf{q}$.

Using B-spline curves, a silhouette is represented by a *point* $\mathbf{q}$ in an *spline space* $\mathbb{R}^{N_q}$. For complex silhouettes, the dimensionality of this space can be considerably big. This is inconvenient for tracking purposes. Estimating a 2D shape would require managing high dimensional state vectors, which would be computationally expensive. Besides, the estimation obtained would be prone to be erroneous: the bigger the number of parameters to be estimated, the higher the probability of wrongly estimating some of them. For this reason, models requiring less parameters are preferred. The most popular approach is based on defining a *shape space* which details a mapping of shape space vectors $\mathbf{c} \in \mathbb{R}^{N_c}$ to spline space vectors $\mathbf{q} \in \mathbb{R}^{N_q}$, where $N_c \ll N_q$. In that way, the feasible values of $\mathbf{q}$ are constrained in a given subspace of $\mathbb{R}^{N_q}$. Next section focuses on two common methods to generate desired shape spaces to describe, respectively, local and global contour transformations.

2.1. Local contour transformations

A simple way to construct a generative model to account for local variations (i.e., deformations) of a mean shape $\bar{\mathbf{q}}$ consists in defining a linear *shape space* $\mathscr{L}(\mathbf{W}^D, \bar{\mathbf{q}})$ of the form

$$\mathbf{q} = \mathbf{W}^D\mathbf{c}^D + \bar{\mathbf{q}}.$$

$\mathbf{W}^D$ is an $N_q \times N_c$ *shape matrix*, $\mathbf{c}^D$ the shape space vector and $\bar{\mathbf{q}}$ the control points of the average contour of the modeled shape. The resultant vector $\mathbf{q}$ corresponds to a linear combination of the columns in $\mathbf{W}^D$ added to $\bar{\mathbf{q}}$. Thus, the family of shape variations represented by $\mathscr{L}(\mathbf{W}^D, \bar{\mathbf{q}})$ depends on the columns of $\mathbf{W}^D$, which are the *basis* of the shape space. To account for the specific variability of a shape of interest, the usual approach is establishing $\mathscr{L}(\mathbf{W}^D, \bar{\mathbf{q}})$ from training data. Given a set $\{\mathbf{q}_i\}_{i=1}^N$ of spline control points of shapes in a training sequence, its mean and covariance $(\bar{\mathbf{q}}, \mathbf{\Sigma})$ are computed by

$$\bar{\mathbf{q}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{q}_i,$$

$$\mathbf{\Sigma} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{q}_i - \bar{\mathbf{q}})\mathscr{M}(\mathbf{q}_i - \bar{\mathbf{q}})^T,$$

where $\mathscr{M}$ is a metric matrix which allows to measure the distance between B-spline curves from only their control points (see [20, Chapter 3]). Performing a principal component analysis on $\mathbf{\Sigma}$, the principal modes of variation of the examples in the training data are obtained as the eigenvectors of $\mathbf{\Sigma}$. The N_c most significant eigenvectors (i.e., the N_c with largest eigenvalues) are used to conform the basis in $\mathbf{W}^D$. N_c is usually established as the minimum number of eigenvectors whose sum of eigenvalues exceeds a given percentage of the total eigenvalues sum. The eigenvectors discarded are considered as accounting for noise in training examples. The interesting point is that $N_c \ll N_q$, and thus a more compact shape model is obtained.

2.2. Global contour transformations

A possibility to construct a shape space that accounts for global transformations consists in designing a matrix $\mathbf{W}^R$ parameterized to account for a given space of similarities, like Euclidean or affine transformations. If $\bar{\mathbf{q}}$ is the outline of a planar target, these spaces synthesize the image projections of this target under specific rigid transformations in 3D. Due to that, in an abuse of language, these shape spaces are commonly said to model rigid contour transformations.

To model the affine transformations of $\bar{\mathbf{q}}$ a 5D shape space can be defined with the expression

$$\mathbf{q} = \mathbf{W}^R(\mathbf{c}^R)\bar{\mathbf{q}} + \mathbf{T}\mathbf{c}^R,$$

where the shape vector $\mathbf{c}^R = [t_x\ t_y\ s_x\ s_y\ \theta]^T$ specifies 2D translation, scaling and rotation of this shape. $\mathbf{W}(\,)$ is a function matrix on $\mathbf{c}^R$ given by

$$\mathbf{W}^R(\mathbf{c}) = \begin{bmatrix} s_x\mathbf{I}_{N_{cp}} & \mathbf{0}_{N_{cp}\times N_{cp}}\\ \mathbf{0}_{N_{cp}} & s_y\mathbf{I}_{N_{cp}\times N_{cp}}\end{bmatrix}\begin{bmatrix}\cos\theta\mathbf{I}_{N_{cp}} & -\sin\theta\mathbf{I}_{N_{cp}}\\ \sin\theta\mathbf{I}_{N_{cp}} & \cos\theta\mathbf{I}_{N_{cp}}\end{bmatrix}, \quad (1)$$

and $\mathbf{T}$ the matrix

$$\mathbf{T} = \begin{bmatrix}\mathbf{1}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \mathbf{0}_{N_{cp}}\\ \mathbf{0}_{N_{cp}} & \mathbf{1}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \mathbf{0}_{N_{cp}}\end{bmatrix}. \quad (2)$$

Notice that this proposal of affine shape space is non-linear. Taking advantage that $\bar{\mathbf{q}}$ is constant along a sequence, an alternative 6D linear shape space $\mathscr{L}(\mathbf{W}, \bar{\mathbf{q}})$ can be defined, with

$$\mathbf{W}^R = \begin{bmatrix}\mathbf{1}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \bar{\mathbf{q}}^x & \mathbf{0}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \bar{\mathbf{q}}^y\\ \mathbf{0}_{N_{cp}} & \mathbf{1}_{N_{cp}} & \mathbf{0}_{N_{cp}} & \bar{\mathbf{q}}^y & \bar{\mathbf{q}}^x & \mathbf{0}_{N_{cp}}\end{bmatrix},$$

where $\bar{\mathbf{q}} = [\bar{\mathbf{q}}^x\ \bar{\mathbf{q}}^y]^T$. Now a spline-vector $\mathbf{q}$ is synthesized with the expression

$$\mathbf{q} = \mathbf{W}^R\mathbf{c}'^R + \bar{\mathbf{q}}. \quad (3)$$

The vector $\mathbf{c}'^R$ relates with the affine parameters in $\mathbf{c}^R$ as

$$\mathbf{c}'^R = [t_x \;\; t_y \;\; (s_x\cos\theta - 1)(s_y\cos\theta - 1) \;\; (s_y\sin\theta) \;\; (-s_x\sin\theta)]^T.$$

2.3. Local and global contour transformations

Preceding sections have modeled separately the local and global transformations of a shape. However, in real applications targets usually display both transformations simultaneously. To account for both in a single parametric model, we can use the expression given in [25]

$$\mathbf{q} = \mathbf{W}^R(\mathbf{c}^R)(\mathbf{W}^D\mathbf{c}^D + \bar{\mathbf{q}}) + \mathbf{T}\mathbf{c}^R, \tag{4}$$

where $\mathbf{c}^D$ maintain the local deformation parameters and $\mathbf{c}^R$ the affine transformation parameters. $\mathbf{W}^R_{()}$ and $\mathbf{T}$ have the same form as in (1) and (2). In order to obtain a linear shape model of the same form as (3), in [25,26] is proposed to expand Eq. (4) by using simple mathematical manipulation, resulting in a shape matrix $\mathbf{W}$ defined as

$$\mathbf{W} = \left[\mathbf{W}^R \begin{bmatrix} \mathbf{W}^D_x & -\mathbf{W}^D_y \\ \mathbf{W}^D_y & -\mathbf{W}^D_x \end{bmatrix}\right],$$

where $\mathbf{W}^D_x$ and $\mathbf{W}^D_y$ correspond to the upper and lower half of $\mathbf{W}^D$, respectively. The parameters of this shape space relate with the affine and local deformation parameters as

$$\mathbf{c}' = [\mathbf{c}'^R \;\; s_x\cos\theta\mathbf{c}^D \;\; s_y\sin\theta\mathbf{c}^D]^T.$$

The drawback of this approach is that it induces linear dependencies between the elements of the parameter vector $\mathbf{c}'$, which is undesirable. The rotation and scaling of the shape are jointly managed by several parameters, that at the same time account for local deformations. Since the proportion of change of this combined transformations can have widely varying magnitudes, this can cause numerical instability in the tracking system. Furthermore, establishing a proper model for the parameter dynamics is also complicated by this fact. To avoid these problems, in this paper we use the shape model as defined in Eq. (4). It has the inconvenience that contours are synthesized through a non-linear expression. However, this does not represent a problem when contours are tracked using a PF. On the other hand, since the model parameters $(\mathbf{c}^D, \mathbf{c}^R)$ act independently, modeling the dynamics of their evolution is far simpler and the interpretation of their values is direct.

3. Contour dynamics

In visual contour tracking applications, the usual approach to describe the expected evolution of the parameters of a given model is by means of an expression based on discrete time series. Given a parameter c, an auto-regressive (AR) process of order n

$$c_t = \sum_{k=1}^{n} \alpha_k c_{t-k} + b_0 w_t$$

is used to describe its dynamics, where α_k are real constants, being $\alpha_k \neq 0$, and $b_0 w_t$ is a stochastic disturbance term corresponding to a Gaussian white noise process with parameters $\mathcal{N}(0, b_0 b_0^T)$.

For mathematical convenience, dynamics are commonly expressed in the vector–matrix form proposed by the state-space notation. In that way, the n-th order dynamics of c_t is described as a first-order Markov process in state space, given by

$$\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{w}_t, \tag{5}$$

where $\mathbf{x}_t = [c_t \;\; c_{t-1} \ldots c_{t-(n-1)}]^T$ is denoted as the system state, $\mathbf{A}$ is the system matrix that defines the deterministic part of the dynamics, and $\mathbf{B}$ is the noise matrix that modulates a Gaussian perturbation vector $\mathbf{w}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. They are, respectively, defined as

$$\mathbf{A} = \begin{bmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_{n-1} & \alpha_n \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} b_0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 \end{bmatrix}.$$

A third parameter $\bar{\mathbf{x}}$ can be added to fix a desired *mean* dynamical behavior, leading to the following expression of dynamics:

$$\mathbf{x}_t - \bar{\mathbf{x}} = \mathbf{A}(\mathbf{x}_{t-1} - \bar{\mathbf{x}}) + \mathbf{B}\mathbf{w}_t.$$

In order to properly set $\mathbf{A}$, $\mathbf{B}$, $\bar{\mathbf{x}}$ for a given problem, different proposals have been done to establish them from training sequences [27–29]. Although learning techniques provide quite accurate descriptions of the behavior observed in these sequences, their use in practical applications may require providing very long and complete training examples. Otherwise, a model very specific to the given data may be obtained (i.e., the overfitting problem), which can be obviously counterproductive. In general, in many applications the a priori knowledge on the dynamics of a process is quite loose, since its evolution can perform a wide spectra of variations. In these cases, instead of determining the AR parameters from training sequences it is better to establish them from statistics of the expected motion. In this paper we follow this second approach to parameterize AR processes of first order (AR1). We use these simple models because in that way the dimension of $\mathbf{x}_t$ is minimized, which, as stated, is interesting when PFs are used. Loose a priori knowledge of parameter dynamics is used to establish the AR process, defining what is known as constrained Brownian motion (CBM) model.

For an AR1, the terms in Eq. (5) correspond to $\mathbf{x_t} = [c_t]$, $\mathbf{A} = [\alpha]$ and $\mathbf{B} = [b_0]$. It can be shown (see [20, Chapter 9]), that if $\alpha^2 = 1 - \varepsilon$ with $0 < \varepsilon \ll 1$ then the dynamics of $\mathbf{x}_t$ resembles a Brownian motion on a small time-scale, but in the long term it is observed that $\mathbf{x}_t$ is confined in a Gaussian envelope given by $\mathcal{N}(0, 1/\varepsilon\mathbf{B}\mathbf{B}^T)$. We propose to take advantage on that to adjust $\mathbf{A}$ and $\mathbf{B}$ in order to account for the lax a priori knowledge about a parameter behavior that is commonly available:

C1 that their values are confined in a given range $[-l, l]$,
C2 that the disturbance term $b_0 w_t$ is expected to take an average magnitude equivalent to a given number m.

By fulfilling these two conditions we control in some way the long and short term parameter evolution. $\mathbf{A}$ and $\mathbf{B}$ are determined taking advantage of the following. It is well known that a random variable with Normal distribution $\mathcal{N}(0, \sigma^2)$ is constrained with a 99.73% probability in the range $[-3\sigma, 3\sigma]$. From this property, the Gaussian envelope in a CBM establishes a range of more likely $\mathbf{x}_t$ values, which can be adjusted to have the desired bounds $[-l, l]$. Thus, a CBM that fulfills constraint C1 requires that

$$l = 3b_0/\sqrt{\varepsilon}. \tag{6}$$

Another property of normal random variables $\mathcal{N}(0, \sigma^2)$ is that the expectation of their absolute value corresponds to $\sigma\sqrt{2/\pi}$. This property can be used to tune the CBM to fulfill C2, since the disturbance term $b_0 w_t$ of an AR process follows a distribution $\mathcal{N}(0, b_0 b_0^T)$. This requires a b_0 value given by

$$b_0 = m\sqrt{\pi/2}. \tag{7}$$

Hence, the more likely disturbance expected m determines the value of b_0. Once b_0 is established, then combining (6) and (7) is found that

$\varepsilon = (9\pi m^2)/(2I^2)$ and the parameter α of the desired AR1 is obtained as $\alpha = \sqrt{1-\varepsilon}$.

In case there exist correlation between parameters, a variant of this approach can be used. Using the eigenvectors of the covariance matrix of correlated parameters, parameters are projected on their diagonalized space, and then a CBM is established there, confined in a range defined by the matrix eigenvalues. Next, this CBM is translated to the original parameter space, just undoing the eigenvector projection.

4. Contour observation model

Once modeled the shape of the target and its expected dynamical behavior, it lacks to clarify how the model relates to the information available of the target to be tracked (i.e., its measurements). This is the task accomplished by the system observation model.

In contour tracking applications, observations usually correspond to salient edges in frames. The typical procedure to extract them is based on synthesizing the contour expected to be found in a frame, and establish several measurement lines normal to this contour, at different points along it. Each measurement line determines a vector of image pixels that are first preprocessed (to filter out image noise, or segment the target of interest according to some discriminant feature) and then an edge detector is applied. The edge closest to the predicted contour is commonly selected, and determines the contour observation at this contour point. This approach is very popular because of their low computational cost, and with respect to alternative approaches, as could be defining observations in terms of spline contour points deducted from image features [26], it is insensible to shape reparameterization (see [20, Chapter 3]).

It is important to notice that the observations obtained with this procedure do not provide the localization of the contour being tracked but just its normal displacement with respect to the contour used to generate the measurement lines. This fact, obviated or misinterpreted in many works, reflects the well-known aperture problem. That is, using just local information, the localization of a contour point in the image is ambiguous. In this case, from the variety of feasible contour displacements that may have occurred, the one normal to the contour is perceived. In other words, we cannot associate a contour point with its corresponding observation, but we can compute the normal component of this association.

Hence, the described measurement process provides the normal displacement between a contour in the image $\mathbf{r}_f$, and a measurement contour $\mathbf{r}_m$ used to establish the measurement lines. To relate these observations with a given shape state $\mathbf{x}$, simply it has to be computed the expected normal displacement between $\mathbf{r}_m$ and the contour $\mathbf{r}_\mathbf{x}$ synthesized from $\mathbf{x}$. If the displacement in both cases match, this means that $\mathbf{r}_f$ and $\mathbf{r}_\mathbf{x}$ overlap. Otherwise, the divergence observed reflects their misalignment (Fig. 2). This leads to the following observation model:

$$\mathbf{y}_T = \mathbf{N}_m^t \mathbf{U}\left(\left(\mathbf{W}^R\left(\mathbf{x}^R\right)\left(\mathbf{W}^D\mathbf{x}^D + \bar{\mathbf{q}}\right) + \mathbf{T}\mathbf{x}^R\right) - \left(\mathbf{W}^R\left(\mathbf{x}_m^R\right)\left(\mathbf{W}^D\mathbf{x}_m^D + \bar{\mathbf{q}}\right) + \mathbf{T}\mathbf{x}_m^R\right)\right) + \mathbf{v},$$

where $\mathbf{U}$ is the matrix that translates spline control points $\mathbf{q}$ onto contour samples, and $\mathbf{N}_m$ is an $N_s \times N_s$ matrix maintaining the normal vectors $[n_x(s_i)\ n_y(s_i)]^T$ at the points sampled along $\mathbf{r}_m$ (synthesized from $\mathbf{x}_m$) as

$$\mathbf{N}_m = \begin{bmatrix} n_x(s_0) & 0 & \dots & 0 \\ 0 & n_x(s_1) & \dots & 0 \\ \vdots & \vdots & \dots & n_x(s_{N_s-1}) \\ n_y(s_0) & 0 & \dots & 0 \\ 0 & n_y(s_1) & \dots & 0 \\ \vdots & \vdots & \dots & n_y(s_{N_s-1}) \end{bmatrix}.$$

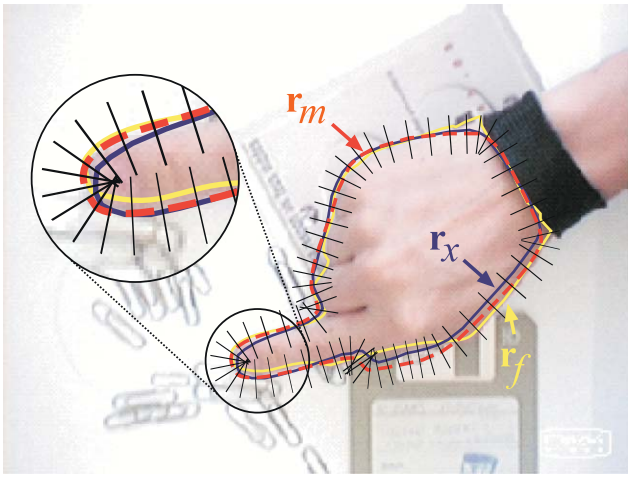

Fig. 2. Contour measurement process.

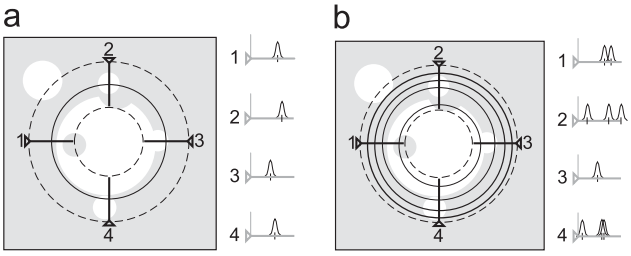

Fig. 3. Inherent likelihood densities for different measurement lines, in Kalman (a) and particle filters (b). The 4 solid circles in (b) show, respectively, the measurement contours $\mathbf{r}_m$ of 4 different particles, which lead to the plotted likelihood densities.

In a Kalman-based filter this measurement extraction process is commonly performed once per image, where $\mathbf{r}_m$ is defined in terms of the Kalman state prediction (i.e., $\mathbf{r}_m = \mathbf{r}_x$ in Fig. 2). The collected observation vector $\mathbf{y}$ is assumed to provide the normal displacement with respect to $\mathbf{r}_f$, except for some Gaussian disturbance. This implicitly means assuming that the state likelihood density $p(\mathbf{y}|\mathbf{x})$ is Gaussian, that there are no more edges in the image than the ones detected (i.e., just the ones closer to $\mathbf{r}_m$), and that these edges correspond effectively to the tracked target (Fig. 3(a)). This is obviously very unrealistic in many application domains. In more formal terms, from all the edge information $\mathbf{y}_t$ available at instant only a small subset $\mathbf{y}_t^m \in \mathbf{y}_t$ is considered in the estimation process.

In a PF the same measurement process is performed, but for each one of the particles considered. The specific measurement process of each particle uses a measurement contour $\mathbf{r}_m$ defined in terms of its own particular state value. Since the edges detected by each particle can correspond to different objects in the scene, this inherently means that the likelihood function $p(\mathbf{y}_t|\mathbf{x}_t)$ can be multimodal (Fig. 3(b)). This is an essential difference with respect to Kalman-based solutions, that explains why PFs can overcome them in cluttered scenarios. An analytic form of the multimodal $p(\mathbf{y}_t|\mathbf{x}_t)$ underlying the PF measurement process is proposed in [30], where not only the presence of cluttering edges is taken into account, but also the possibility of contour occlusions or misdetections.

5. Contour tracking using importance sampling

The goal of PFs is to generate a particle set representing properly the distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. This density is just the marginal of $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$, which we use in the following to formalize PFs more

easily. Expressions derived to properly represent $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ will necessarily characterize $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. Hence, the task to be fulfilled is determining a particle set that should correspond to a random sampling of $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$. Since this density is unknown, generating samples from it is not possible. However, we can take advantage of the importance sampling technique [31] to approximate it. This method requires that the following assumptions hold:

- Although $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ cannot be sampled, it is possible to evaluate it for a concrete $\mathbf{x}_{0:t}$ up to a constant factor.
- There exists an importance distribution $q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ that can be sampled, and also evaluated up to a constant factor. This distribution is such that $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})>0$ implies $q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})>0$.

Very roughly, the procedure carried out is the following: a sample set $\{\mathbf{x}_{0:t}^{(i)}\}_{i=1}^{N}$ is first generated from $q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$. Then, a weight $\tilde{w}_t^{(i)}$ is associated to each sample $\mathbf{x}_{0:t}^{(i)}$, which takes into account if $\mathbf{x}_{0:t}^{(i)}$ can be considered really a sample of $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$. The desired posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ is finally approximated by

$$\widetilde{p_N}(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \sum_{i=1}^{N} \delta_{\mathbf{x}_{0:t}^{(i)}} \tilde{w}_t^{(i)}, \tag{8}$$

where $\delta_{\mathbf{x}_{0:t}^{(i)}}$ is the Dirac delta function centered at $\mathbf{x}_{0:t}^{(i)}$, which is a random sample generated from $q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$, and

$$\tilde{w}_t^{(i)} = w_t^{(i)} \Big/ \sum_{j=1}^{N} w_t^{(j)},$$

$$w_t^{(i)} = \frac{p(\mathbf{x}_{0:t}^{(i)}|\mathbf{y}_{1:t})}{q(\mathbf{x}_{0:t}^{(i)}|\mathbf{y}_{1:t})}.$$

The weights $\tilde{w}_t$ and w_t are denoted, respectively, as normalized and unnormalized importance weights. If the importance function is chosen of the form

$$q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = q(\mathbf{x}_0)\prod_{k=1}^{t} q(\mathbf{x}_k|\mathbf{x}_{0:k-1}\mathbf{y}_{1:k}),$$

the weighted sample set in (8) can be estimated efficiently along time by a recursive expression. Using Bayes' theorem, the importance weight w_t turns out to be

$$\begin{aligned} w_t &= \frac{p(\mathbf{y}_t|\mathbf{x}_{0:t}\mathbf{y}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t-1})p(\mathbf{x}_{0:t-1}|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})} \\ &= \frac{p(\mathbf{y}_t|\mathbf{x}_{0:t}\mathbf{y}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})q(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t})}\frac{p(\mathbf{x}_{0:t-1}|\mathbf{y}_{1:t-1})}{q(\mathbf{x}_{0:t-1}|\mathbf{y}_{1:t-1})} \\ &= \frac{p(\mathbf{y}_t|\mathbf{x}_{0:t}\mathbf{y}_{1:t-1})p(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})q(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t})}w_{t-1}. \end{aligned} \tag{9}$$

Thus, the importance weight of a particle at instant t can be computed in terms of its preceding value. In many formulations, the term $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ is omitted from (9), since acts as a normalization factor which cancels when $\tilde{w}_t$ is computed. In most practical cases, since first-order state dynamics are used and observations are independently conditioned on $\mathbf{x}_t$, the weight w_t is finally computed as

$$w_t \propto \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})}{q(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t})}w_{t-1}. \tag{10}$$

This procedure conforms what is denoted as the sequential importance sampling (SIS) algorithm, which consists in the recursive propagation of weights and samples of $\widetilde{p_N}(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ as each new measurement is received. The initial particle set $\{\mathbf{x}_0^{(i)}, w_0^{(i)}\}_{i=1}^{N}$ propagated by this algorithm is determined from system priors, so that

$$\mathbf{x}_0^{(i)} \sim q(\mathbf{x}_0) = p(\mathbf{x}_0),$$
$$w_0^{(i)} = 1.$$

However, if this procedure is applied, it can be proved [14,32] that at each iteration the variance of weights can only increase, which in practical terms means that after several iterations, all but one particle will have negligible weight. This provokes that a large computation effort is devoted to updating particles whose contribution to $\widetilde{p_N}(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ is almost zero, which consequently represents a poor approximation of $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$. This is commonly denoted as the *degeneracy* phenomenon. An operative solution to this problem did not appear until the last decade, when in [33] was proposed the addition of a *resampling* step at the end of the SIS iteration. The intuitive idea is to get rid of particles with small weights, focusing on particles of bigger *importance*. This is achieved by resampling with replacement the point mass distribution $\widetilde{p_N}(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$, generating a (theoretically) equivalent point distribution density given by

$$\widehat{p_N}(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\mathbf{x}_{0:t}^{(i)}}.$$

With this process, the importance of each particle now gets reflected in the number of copies generated from it, having each copy a weight equal to $1/N$. This obviously resets the particle weight variance to the feasible minimum. To implement the resampling step, several algorithms have been proposed, being popular multinomial [33], stratified [34], residual [35] and systematic [36] resampling schemes. A comparison between them can be found in [37].

This resampling step is applied by some authors at each SIS iteration. However, since its finality is addressing the degeneracy problem, it is reasonable to apply it only when this degeneracy effectively is present on the particle set. A suitable measure of degeneracy is the *effective sample size* N_{eff} introduced in [32], a term commonly estimated (see [5,14,30]) using the approximation

$$N_{eff} \sim 1 \Big/ \sum_{i=1}^{N} (\tilde{w}_t^{(i)})^2.$$

In the ideal non-degenerated case, all particles have identical weight $1/N$, resulting that $N_{eff}=N$. As the weight variance increases, N_{eff} diminishes. Thus, a common approach is applying the resampling step only when N_{eff} is below a defined threshold N_{deg}. Fig. 4 specifies the general framework of sequential importance sampling with resampling (SISR) algorithms, popularly denoted as PFs, and Fig. 5 details the resampling step.

$[\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^{N}] = \mathrm{SISR}[\{\mathbf{x}_{0:t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}, \mathbf{y}_{1:t}]$
for $i = 1$ to N **do**
Draw $\mathbf{x}_t^{(i)} \sim q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t})$
Set $\mathbf{x}_{0:t}^{(i)} = [\mathbf{x}_{0:t-1}^{(i)}\ \mathbf{x}_t^{(i)}]$
Update weight
$w_t^{(i)} = \frac{p(\mathbf{y}_t|\mathbf{x}_{0:t}^{(i)}\mathbf{y}_{1:t-1})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t-1})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t})}w_{t-1}^{(i)}$
end for
Resampling Step
$[\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^{N}] = \mathrm{R}[\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^{N}]$

Fig. 4. Sequential importance sampling with resampling algorithm.

$[\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^N] = \mathrm{R}[\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^N]$

Normalise weights

$\tilde{w}_t^{(i)} = w_t^{(i)} \Big/ \sum_{j=1}^N w_t^{(j)} \quad \forall i = 1, \ldots, N$

Compute $N_{eff} = 1 \Big/ \sum_{i=1}^N (\tilde{w}_t^{(i)})^2$

if $N_{eff} < N_{deg}$ **then**

Resample particle set with selected algorithm A

$[\{\mathbf{x}_{0:t}^{(i)}, w_t^{(i)}\}_{i=1}^N] = \mathrm{A}[\{\mathbf{x}_{0:t}^{(i)}, \tilde{w}_t^{(i)}\}_{i=1}^N]$

end if

Fig. 5. Resampling step.

$[\{\mathbf{x}_{0:t}^{(i)}, \Sigma_{0:t}^{\mathbf{xx}(i)}, w_t^{(i)}\}_{i=1}^N] = \mathrm{UPF}[\{\mathbf{x}_{0:t-1}^{(i)}, \Sigma_{0:t-1}^{\mathbf{xx}(i)}, w_{t-1}^{(i)}\}_{i=1}^N, \mathbf{y}_{1:t}]$

for $i = 1$ to N **do**

$\left(\hat{\mathbf{x}}_t^{(i)}, \Sigma_t^{\mathbf{xx}(i)}\right) = \mathrm{UKF}\left[\left(\mathbf{x}_{t-1}^{(i)}, \Sigma_{t-1}^{\mathbf{xx}(i)}\right), \mathbf{y}_{1:t}\right]$

Set $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t}) = \mathcal{N}(\hat{\mathbf{x}}_t^{(i)}, \Sigma_t^{\mathbf{xx}^{(i)}})$

Draw $\mathbf{x}_t^{(i)} \sim q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t})$

Set $\mathbf{x}_{0:t}^{(i)} \triangleq \left[\mathbf{x}_{0:t-1}^{(i)}\ \mathbf{x}_t^{(i)}\right]$ and $\Sigma_{0:t}^{\mathbf{xx}(i)} \triangleq \left[\Sigma_{0:t-1}^{\mathbf{xx}(i)}\ \Sigma_t^{\mathbf{xx}(i)}\right]$

Update weight $w_t^{(i)} = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t})} w_{t-1}^{(i)}$

end for

Resampling Step

$[\{\mathbf{x}_{0:t}^{(i)}, \Sigma_{0:t}^{\mathbf{xx}(i)}, w_t^{(i)}\}_{i=1}^N] = \mathrm{R}[\{\mathbf{x}_{0:t}^{(i)}, \Sigma_{0:t}^{\mathbf{xx}(i)}, w_t^{(i)}\}_{i=1}^N]$

Fig. 6. Unscented particle filter.

In many tracking applications, the importance function used in the SIRS algorithm is based on the prior distribution of the tracked state (i.e., $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$). This particular implementation, in which particles are propagated along frames by the expected system dynamics model, is commonly referred as Bootstrap filter or Condensation algorithm. From that, the update of the weight of particles in (10) reduces just to

$$w_t \propto p(\mathbf{y}_t|\mathbf{x}_t) w_{t-1},$$

requiring only to evaluate the likelihood of observations.

This algorithm has lead to good tracking results in many application domains, but it has the drawback that if the dimension of $\mathbf{x}_t$ is big, a very big amount of particles is required for a robust performance. In fact, this is a problem inherent of the importance sampling technique, made more evident in the Condensation algorithm because of the *poor* importance function used. The basic problem is that since to represent $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$ we use samples from a proposed $q(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$, it can be shown [31] that with the dimensionality of $\mathbf{x}_t$, the variance of the weights associated to particles increases exponentially. This means that the bigger the state dimension, the smaller the number of effective particles (i.e., with non-negligible weight w_t) representing the estimated distribution. As a consequence, an importance sampling estimate is totally dominated by just a small percentage of samples. The brute force to face this problem consists in using a huge amount of particles. In the following sections we explore three more intelligent strategies, adapting them to contour tracking based on ASM. Their goal is reducing the variance of the weight of particles at each iteration, which implies reducing the amount of useless particles generated. Due to that, they are commonly referred as variance reduction techniques. The techniques analyzed pursue this goal by:

- using a better importance function in the SISR algorithm, considering current observations in their definition (Section 6),
- estimating part of $\mathbf{x}_t$ analytically, and the remaining part, necessarily of lower dimensionality, by means of a particle set (Section 7), and
- taking advantage that in some contour tracking applications, part of the parameters in $\mathbf{x}_t$ can be estimated independently of other parameters (Section 8). This can be used to guide importance samples to zones of high likelihood according to observations.

6. Unscented particle filter

The key point of an SISR algorithm is the importance function $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t})$ used to generate particles at each time step. In [14] it is proved that the optimal function to carry this task (in terms of minimizing the variance of the particle weights w_t) is $p(\mathbf{x}_t|\mathbf{x}_{0:t-1}\mathbf{y}_{1:t})$. Note that this optimal importance sampling density (OISD) is conditioned on current observations. In practical applications, its determination is commonly a non-trivial task. The unscented particle filter [38] proposes as solution approximating this function locally around each particle $\mathbf{x}_{0:t-1}^{(i)}$ by means of a Gaussian distribution. In concrete, for each particle $\mathbf{x}_{0:t-1}^{(i)}$, an unscented Kalman filter (UKF) is used to generate and propagate a Gaussian approximation of the OISD

$$q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t}) = \mathcal{N}(\hat{\mathbf{x}}_t^{(i)}, \boldsymbol{\Sigma}_t^{\mathbf{xx}^{(i)}}).$$

In practice this means that for each particle $(\mathbf{x}_{0:t}^{(i)}, w_{1:t}^{(i)})$, now it is also required to maintain matrices $\boldsymbol{\Sigma}_{0:t}^{\mathbf{xx}^{(i)}}$ detailing the covariance of the associated Gaussian. Fig. 6 details our adaptation of this strategy for the described contour tracking problem.

Following this pseudocode we see that the execution of an UKF is required for each sample, resulting in a high computational cost. However, in practice one can take advantage of a remarkable fact of PFs applied to contour tracking applications. At each iteration, since the contour likelihood $p(\mathbf{x}_t|\mathbf{y}_t)$ is a remarkably narrow function, the amount of particles surviving the resampling step is commonly small (in most cases around the 20% of the total of particles). Due to this fact, at the time of propagating particles most of them are identical, and can share the same approximation of the OISD. Thus, at each iteration it has to be evaluated just as many UKFs as different surviving particles, reducing a great deal the computational cost of the algorithm.

6.1. Related approaches

The application of the UPF in visual contour tracking applications has been also proposed by other authors. However, their approaches differ significantly from ours. In [39] a face tracker is presented, based on modeling the face outline by means of an ellipse of fixed size and orientation. The system only accounts for translations of the contour model. In [40] a contour tracker is proposed, aimed at the estimation of affine transformations of a rigid contour model. These two approaches apply unnecessarily the UPF on their tracking problem, because as long as they use linear system and observation models, a Kalman filter (KF) is enough to approximate the optimal importance function for each particle. In addition, both approaches derive from a wrong model of the contour measurement process. They assume that the disparity *coordinate-by-coordinate* between predicted and observed contours is measured in the image, while in practice only the normal disparity with respect to a defined measurement contour is really obtained. In our proposal, the application of the UPF is completely necessary, since the contour model used is non-linear. Moreover, a better modeling of the observation process is done,

$$\left[\left(\hat{\mathbf{x}}_{t|t}, \Sigma^{\mathbf{xx}}_{t|t}\right)\right] = \mathrm{UKF}\left[\left(\hat{\mathbf{x}}_{t-1|t-1}, \Sigma^{\mathbf{xx}}_{t-1|t-1}\right), \mathbf{y}_{1:t}\right]$$

Time update equations:

$$\hat{\mathbf{x}}_{t|t-1} = \mathbf{A}\mathbf{x}_{t-1|t-1}$$

$$\Sigma^{\mathbf{xx}}_{t|t-1} = \mathbf{A}\Sigma^{\mathbf{xx}}_{t-1|t-1}\mathbf{A}^T + \mathbf{Q}_t$$

Frame observations extraction

$$\mathbf{x}_m = \hat{\mathbf{x}}_{t|t-1}$$

$$(\mathbf{y}^m_t, \mathbf{R}_t) = \mathrm{OBSERVATION\_PROCESS}(\mathbf{x}_m, \mathbf{y}_t)$$

Measurement update equations:

$$\left[\left\{\mathbf{x}^{(i)}_{t|t-1}, w^{(i)}_m, w^{(i)}_c\right\}_{i=1}^{N}\right] = \mathrm{GET\_}\sigma\mathrm{\_POINTS}\left[\left(\hat{\mathbf{x}}_{t|t-1}, \Sigma^{\mathbf{xx}}_{t|t-1}\right)\right]$$

$$\mathbf{y}^{(i)}_{t|t-1} = \mathbf{N}^t_m\mathbf{U}\left(\left(\mathbf{W}^R(\mathbf{x}^{R(i)}_{t|t-1})\ \left(\mathbf{W}^D\mathbf{x}^{D(i)}_{t|t-1} + \bar{\mathbf{q}}\right) + \mathbf{T}\mathbf{x}^{R(i)}_{t|t-1}\right) - \right.$$
$$\left. - \left(\mathbf{W}^R(\mathbf{x}^R_m)\ \left(\mathbf{W}^D\mathbf{x}^D_m + \bar{\mathbf{q}}\ + \mathbf{T}\mathbf{x}^R_m\right)\right)\right)$$

$$\mathbf{y}_{t|t-1} = \textstyle\sum_{i=1}^{N} w^{(i)}_m \mathbf{y}^{(i)}_{t|t-1}$$

$$\Sigma^{\mathbf{yy}}_{t|t-1} = \left(\textstyle\sum_{i=1}^{N} w^{(i)}_c (\mathbf{y}^{(i)}_{t|t-1} - \bar{\mathbf{y}}_{t|t-1})(\mathbf{y}^{(i)}_{t|t-1} - \bar{\mathbf{y}}_{t|t-1})^T\right) + \mathbf{R}_t$$

$$\Sigma^{\mathbf{xy}}_{t|t-1} = \textstyle\sum_{i=1}^{N} w^{(i)}_c (\mathbf{x}^{(i)}_{t|t-1} - \bar{\mathbf{x}}_{t|t-1})(\mathbf{y}^{(i)}_{t|t-1} - \bar{\mathbf{y}}_{t|t-1})^T$$

$$\mathbf{K} = \Sigma^{\mathbf{xy}}_{t|t-1}(\Sigma^{\mathbf{yy}}_{t|t-1})^{-1}$$

$$\hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}(\mathbf{y}^m_t - \bar{\mathbf{y}}_{t|t-1})$$

$$\Sigma^{\mathbf{xx}}_{t|t} = \Sigma^{\mathbf{xx}}_{t|t-1} - \mathbf{K}\Sigma^{\mathbf{yy}}_{t|t-1}\mathbf{K}^T$$

Fig. 7. Unscented Kalman filter.

interpreting rigorously the evidence extracted from processed frames. Details of the specific UKF used in the UPF proposed are shown in Fig. 7, in accordance with the observation model described in Section 4.

The use of Kalman-based filters to approximate the OISD was first suggested in [14]. In this paper the authors also proposed an alternative method to do that, based on computing the OISD Laplace's approximation [41]. Following this approach, in [42] a generalization of this strategy has been recently proposed, denoted as particle filtering with efficient importance sampling (PF-EIS). This work shows that in some situations approximating the OISD with a Gaussian is improper, since it is in fact multimodal. However, when this happens, there are cases where the OISD of part of the state is indeed unimodal conditioned on the remaining part of the state. Hence, they propose to split $\mathbf{x}$ into two parts: the *multimodal* and the *unimodal* part. The multimodal part is sampled using transition priors, as regular PFs do. Conditioned on these samples, now the OISD of the other part can be correctly taken as Gaussian. By applying the Gaussian OISD approximation only in this state partition the effectiveness of the strategy is increased.

A proposal that is close in philosophy with the UPF is the Turbo PF [43]. It is a sort of simplified version of the UPF, where at each time instant a unique Gaussian OISD is defined to generate all the importance samples. Their implementation is based on the cooperation of an extended KF with a PF, whose relationship mimics the one of master and slave filters in Turbo decoding.

Besides the Gaussian approximation strategy, other heuristic methods have been proposed to use recent observations to characterize the OISD. Given $\mathbf{y}_t$, in [43] is proposed to use the gradient descent direction of the measurement model to define an importance function that move particles close to it. Other authors propose not using $\mathbf{y}_t$ directly, but some other image information heuristically correlated with it. For instance, in [44] the pixel-based interframe motion $\mathbf{I}_{o:t}$ is used as a clue for target movement. Using this information, their approach consist in defining the state transition model as $p(\mathbf{x}_t \| \mathbf{x}_{t-1}\mathbf{I}_{0:t})$, which in turn is used to generate samples. Similarly, in [45] this same idea is applied, but just to define the OISD approximation, keeping the state transition model based just on priors.

7. Rao-Blackwellized particle filter

The strategy considered in this section is based on factorizing the desired density $p(\mathbf{x}_{0:t}|\mathbf{y}_{1:t})$. If the state to be estimated is divided into two parts $\mathbf{x}_{0:t} = [\mathbf{x}^{P1}_{0:t}\mathbf{x}^{P2}_{0:t}]^T$, then this density can be factored as

$$p(\mathbf{x}^{P1}_{0:t}\mathbf{x}^{P2}_{0:t}|\mathbf{y}_{1:t}) = p(\mathbf{x}^{P1}_{0:t}|\mathbf{y}_{1:t})p(\mathbf{x}^{P2}_{0:t}|\mathbf{x}^{P1}_{0:t}\mathbf{y}_{1:t}). \tag{11}$$

The Rao-Blackwellization (RB) technique [14,46] proposes to use this structural information to infer analytically a part of the state ($\mathbf{x}^{P2}_{0:t}$) conditionally upon the other part of the state ($\mathbf{x}^{P1}_{0:t}$), which is estimated, respectively, by a sequential Monte Carlo algorithm. In the concrete case of (11), this means solving:

- $p(\mathbf{x}^{P2}_{0:t}|\mathbf{x}^{P1}_{0:t}\mathbf{y}_{1:t})$ analytically;
- $p(\mathbf{x}^{P1}_{0:t}|\mathbf{y}_{1:t})$ by means of a PF.

From that, an estimation of the posterior density $p(\mathbf{x}^{P1}_{0:t}\mathbf{x}^{P2}_{0:t}|\mathbf{y}_{1:t})$ is obtained, given by the following mixture of densities

$$\widetilde{p_N}(\mathbf{x}_{0:t}|\mathbf{y}_{1:t}) = \sum_{i=1}^{N} \tilde{w}^{(i)}_t p(\mathbf{x}^{P2}_{0:t}|\mathbf{x}^{P1(i)}_{0:t}\mathbf{y}_{1:t}). \tag{12}$$

The RB approach states that if the analytic solution to $p(\mathbf{x}^{P2}_{0:t}|\mathbf{x}^{P1}_{0:t}\mathbf{y}_{1:t})$ represents precisely the inherent real distribution, then the proposed methodology directly leads to overcome the accuracy of a *classical* PF approach, since the variance of the particle weights is reduced and a bigger particle diversity is maintained after resampling [46].

Filters using this strategy are commonly denoted as Rao-Blackwellized particle filters, and have been applied in many different areas and problems: jump Markov linear systems [50], neural networks [47], map learning [48], positioning and navigation [49], surveillance [51], multiple target tracking [52], template tracking using rectangular [53] or articulated models [54], etc. In this paper we contribute with the novel adaption of this technique to the contour tracking problem.

Behind the RB concept lies a rule of thumb in estimation theory: if something can be solved analytically, do not solve it by means of sampling techniques. In our case, to obtain an analytic solution for $p(\mathbf{x}^{P2}_{0:t}|\mathbf{x}^{P1}_{0:t}\mathbf{y}_{1:t})$ we use a KF. This inherently means modeling the estimation of $\mathbf{x}^{P2}$ given $\mathbf{x}^{P1}$ by means of linear Gaussian processes. In the problem of visual contour tracking, estimating part of the state analytically from $\mathbf{y}_{1:t}$ is in general bad posed, as non-Gaussian artifacts can distort observations. Thus, the benefits of applying the RB technique onto this application cannot be taken for granted a priori. To ascertain if there are more advantages than disadvantages in using this technique, we have evaluated its performance experimentally, providing quantitative insight onto this point (see Section 9). Let us first detail the Rao–Blackwellized estimation of $p(\mathbf{x}^{P1}_{0:t}\mathbf{x}^{P2}_{0:t}|\mathbf{y}_{1:t})$. It requires updating at each time t the distribution in Eq. (12), which implies:

- updating the sample distribution of $p(\mathbf{x}^{P1}_{0:t}|\mathbf{y}_{1:t})$ given by a weighted sample set $\{(\mathbf{x}^{P1(i)}_{0:t}, \tilde{w}^{(i)}_t)\}_{i=1}^{N}$;
- updating for each i-th sample $\mathbf{x}^{P1(i)}_{0:t}$, the distribution $p(\mathbf{x}^{P2}_{0:t}|\mathbf{x}^{P1(i)}_{0:t}\mathbf{y}_{1:t})$. This is directly solved using a KF, which estimates the parameters of the normal density $\mathcal{N}(\hat{\mathbf{x}}^{P2(i)}_{0:t}, \boldsymbol{\Sigma}^{\mathbf{x}^{P2}\mathbf{x}^{P2}}_{0:t})$.

$$[\{\mathbf{x}_{0:t}^{P1(i)}, (\hat{\mathbf{x}}_{0:t}^{P2(i)}, \Sigma_{0:t}^{\mathbf{x}^{P2}\mathbf{x}^{P2}(i)}), w_t^{(i)}\}_{i=1}^{N}] = \mathrm{RBPF}[\{\mathbf{x}_{0:t-1}^{P1(i)}, (\hat{\mathbf{x}}_{0:t-1}^{P2(i)}, \Sigma_{0:t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}(i)}), w_{t-1}^{(i)}\}_{i=1}^{N}, \mathbf{y}_{1:t}]$$

for $i = 1$ to N **do**

Draw $\mathbf{x}_t^{P1(i)} \sim p(\mathbf{x}_t^{P1} \; \mathbf{x}_{t-1}^{P1(i)})$

Kalman Prediction Step

$$\hat{\mathbf{x}}_{t|t-1}^{P2} = \mathbf{A}^{P2}\mathbf{x}_{t-1}^{P2(i)}$$

$$\Sigma_{t|t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}} = \mathbf{A}^{P2}\Sigma_{t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}(i)}\mathbf{A}^{P2^T} + \mathbf{Q}_t^{P2}$$

Kalman Update Step

$$\mathbf{x}_m = \left[\mathbf{x}_t^{P1(i)} \; \hat{\mathbf{x}}_{t|t-1}^{P2}\right] \quad \left\{ \text{ but only } \mathbf{x}_m^{P1} == \mathbf{x}_t^{P1(i)} \text{ is really required} \right\}$$

$$(\mathbf{y}_t^m, \mathbf{R}_t) = \mathrm{OBSERVATION\_PROCESS}(\mathbf{x}_m, \mathbf{y}_t)$$

$$\mathbf{K} = \frac{\Sigma_{t|t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}}\mathbf{H}_{|\mathbf{x}_t^{P1(i)}}^T}{\mathbf{H}_{|\mathbf{x}_t^{P1(i)}}\Sigma_{t|t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}}\mathbf{H}_{|\mathbf{x}_t^{P1(i)}}^T + \mathbf{R}_t}$$

$$\nu_t = \mathbf{y}_t^m - \mathbf{H}_{|\mathbf{x}_t^{P1(i)}}\left(\hat{\mathbf{x}}_{t|t-1}^{P2} - \mathbf{x}_m^{P2}\right)$$

$$\hat{\mathbf{x}}_{t|t}^{P2} = \hat{\mathbf{x}}_{t|t-1}^{P2} + \mathbf{K}\nu_t$$

$$\Sigma_{t|t}^{\mathbf{x}^{P2}\mathbf{x}^{P2}} = \left(\mathbf{I} - \mathbf{K}\mathbf{H}_{|\mathbf{x}_t^{P1(i)}}\right)\Sigma_{t|t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}}$$

Set $\mathbf{x}_{0:t}^{P1(i)} \triangleq \left[\mathbf{x}_{0:t-1}^{P1(i)} \; \mathbf{x}_t^{P1(i)}\right]$

Set $\hat{\mathbf{x}}_{0:t}^{P2(i)} \triangleq \left[\hat{\mathbf{x}}_{0:t-1}^{P2(i)} \; \hat{\mathbf{x}}_{t|t}^{P2}\right]$

Set $\Sigma_{0:t}^{\mathbf{x}^{P2}\mathbf{x}^{P2}(i)} \triangleq \left[\Sigma_{0:t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}(i)} \; \Sigma_{t|t}^{\mathbf{x}^{P2}\mathbf{x}^{P2}}\right]$

Update weight

$$w_t^{(i)} = \Phi\left(\nu_t, \mathbf{H}_{|\mathbf{x}_t^{P1(i)}}\Sigma_{t|t-1}^{\mathbf{x}^{P2}\mathbf{x}^{P2}}\mathbf{H}_{|\mathbf{x}_t^{P1(i)}}^T + \mathbf{R}_t\right) w_{t-1}^{(i)}$$

where $\Phi(\mu, \Sigma) = \frac{1}{|2\pi\Sigma|^{1/2}} e^{-\frac{1}{2}\;{}^T\Sigma^{-1}}$

end for

Resampling Step

$$[\{\mathbf{x}_{0:t}^{P1(i)}, (\hat{\mathbf{x}}_{0:t}^{P2(i)}, \Sigma_{0:t}^{\mathbf{x}^{P2}\mathbf{x}^{P2}(i)}), w_t^{(i)}\}_{i=1}^{N}] = \mathrm{R}[\{\mathbf{x}_{0:t}^{P1(i)}, (\hat{\mathbf{x}}_{0:t}^{P2(i)}, \Sigma_{0:t}^{\mathbf{x}^{P2}\mathbf{x}^{P2}(i)}), w_t^{(i)}\}_{i=1}^{N}]$$

Fig. 8. Rao-Blackwellized particle filter.

The estimation of $p(\mathbf{x}_{0:t}^{P1}|\mathbf{y}_{1:t})$ is solved using the SIS algorithm. Now, samples $\mathbf{x}_{0:t}^{P1(i)}$ are generated from an importance function $q(\mathbf{x}_{0:t}^{P1}|\mathbf{y}_{1:t})$ following the form:

$$q(\mathbf{x}_{0:t}^{P1}|\mathbf{y}_{1:t}) = q(\mathbf{x}_0^{P1})\prod_{k=1}^{t} q(\mathbf{x}_k^{P1}|\mathbf{x}_{0:k-1}^{P1}\mathbf{y}_{1:k}).$$

Using this type of importance function, the unnormalized weight factor w_t associated to each particle $\mathbf{x}_{0:t}^{P1(i)}$ is determined recursively as

$$w_t^{(i)} \propto \frac{p(\mathbf{y}_t|\mathbf{y}_{1:t-1}\mathbf{x}_{0:t}^{P1(i)})p(\mathbf{x}_t^{P1(i)}|\mathbf{x}_{t-1}^{P1(i)})}{q(\mathbf{x}_t^{P1(i)}|\mathbf{x}_{0:t-1}^{P1(i)}\mathbf{y}_{1:t})} w_{t-1}^{(i)}.$$

If the prior distribution $p(\mathbf{x}_t^{P1}|\mathbf{x}_{t-1}^{P1})$ is used as importance distribution, then w_t is proportional to

$$w_t \propto p(\mathbf{y}_t|\mathbf{y}_{1:t-1}\mathbf{x}_{0:t}^{P1})w_{t-1}.$$

Thus, the unnormalized weight $w_t^{(i)}$ requires just to evaluate $p(\mathbf{y}_t|\mathbf{y}_{1:t-1}\mathbf{x}_{0:t}^{P1(i)})$. The nice point of the Rao-Blackwellization strategy is that this term corresponds to the evaluation of $\mathbf{y}_t$ in the one-step-ahead Kalman prediction of the observation density computed when $p(\mathbf{x}_{0:t}^{P2}|\mathbf{x}_{0:t}^{P1(i)}\mathbf{y}_{1:t})$ is solved. This density corresponds to

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}\mathbf{x}_{0:t}^{P1(i)}) \sim \mathcal{N}(\hat{\mathbf{y}}_{t|t-1}, \boldsymbol{\Sigma}_{t|t-1}^{\mathbf{yy}}),$$

where $\hat{\mathbf{y}}_{t|t-1}$ and $\boldsymbol{\Sigma}_{t|t-1}^{\mathbf{yy}}$ are the mean and covariance of the observations expected from a contour with B-spline contour points corresponding to the joint state $[\mathbf{x}_t^{P1(i)} \; \hat{\mathbf{x}}_{t|t-1}^{P2}]^T$.

Fig. 8 details a generic implementation of the RBPF proposed. In practice, this pseudocode corresponds just to an iterative method to update a Gaussian mixture model in $\widetilde{p_N}(\mathbf{x}_{0:t}^{P1}\mathbf{x}_{0:t}^{P2}|\mathbf{y}_{0:t})$. In this mixture, one could interpret that several hypothesis on the sequence of $\mathbf{x}^{P1}$ values are maintained, and for each one of them, a KF is used to estimate the posterior on $\mathbf{x}^{P2}$. Notice that $\mathbf{x}^{P1}$ and $\mathbf{x}^{P2}$ require uncorrelated dynamical processes, since we need to sample from $p(\mathbf{x}_t^{P1}|\mathbf{x}_{t-1}^{P1(i)})$.

In our specific application of the RBPF in the contour tracking problem, the state partitions $[\mathbf{x}^{P1} \; \mathbf{x}^{P2}]$ isolate the shape parameters with a non-linear/linear relation with the contour observations (i.e., $\mathbf{x}^{P1} = [s_x \; s_y \; \theta]^T$ and $\mathbf{x}^{P2} = [t_x \; t_y \; \mathbf{x}^D]^T$, respectively). If we use a measurement contour $\mathbf{x}_m = [\mathbf{x}_m^{P1} \; \mathbf{x}_m^{P2}]$ so that $\mathbf{x}_m^{P1}$ is equal to $\mathbf{x}^{P1}$, then for a fixed $\mathbf{x}^{P1}$ the relation with the observations is given by

$$\mathbf{y}^m = \mathbf{H}_{|\mathbf{x}^{P1}}(\mathbf{x}^{P2} - \mathbf{x}_m^{P2}),$$

where

$$\mathbf{H}_{|\mathbf{x}^{P1}} = \mathbf{N}_m^T\mathbf{U}\left[\begin{bmatrix}\mathbf{1}_{N_{cp}} & \mathbf{0}_{N_{cp}} \\ \mathbf{0}_{N_{cp}} & \mathbf{1}_{N_{cp}}\end{bmatrix}\mathbf{W}^R(\mathbf{x}^{P1})\mathbf{W}^D\right].$$

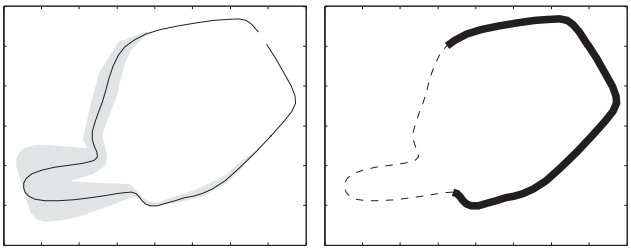

Fig. 9. Almost rigid contour zone. Left: graphical representation of the contour representability of a shape space $\mathscr{L}(\mathbf{W}^D, \bar{\mathbf{q}})$ modeling a hand with a pointing finger. Right: the bold line marks the contour zones that vary minimally for any valid $\mathbf{x}^D$ parameterization, and can approximately be assumed as rigid.

7.1. Related approaches

RBPFs are also referred as marginalized particle filters. Under this denomination, it is described in [55] their particular general form in three different types of nonlinear state-space models with linear substructure: diagonal, triangular, and general models. The model considered in this section is an example of the diagonal case. Close related with RBPF are PF with mode tracking (PF-MT) [42]. Their strategy is in essence the same: making a more efficient use of particles by applying them just to part of the state. The difference is that while RBPFs estimate a Gaussian distribution for the rest of the state, PF-MT just estimates the mode of its underlying distribution, which has not to be necessarily Gaussian. No linear substructure is required in the tracking problem, just unimodality in the posterior of the non-sampled state part. Taking that into account, PF-MT can be seen as a simplification of RBPFs that generalizes their applicability. Note that in PF-MT the state partition is not predetermined beforehand from the system linear substructure, but it can be established dynamically at each time instant. Examples on the application of PF-MT are found in the context of template tracking across illumination changes [56] and in level set-based contour tracking [57].

8. Partitioned sampling

Our third proposal to deal with the curse of dimensionality problem of PFs arises from the following observation: in most real contour tracking applications, the dynamics of global and local transformations are independent, and therefore they can be modeled separately. Could it be possible to estimate also them separately? In general, the answer to this question is clearly no, as observations manifest jointly the effect of both transformations. However, paying attention to specific contour tracking applications, it appears that in most cases a quite accurate estimation of the object global transformation $\mathbf{x}^R$ may be obtained independently of its local deformation $\mathbf{x}^D$. Indeed, commonly modeled objects deform just at localized regions of their outline, and the rest of the outline can be assumed as rigid (see Fig. 9). Thus, changes observed in these rigid regions will be caused only by global contour transformations, and this can be used to estimate $\mathbf{x}^R$, whichever the parameters in $\mathbf{x}^D$ are.

The estimation of $\mathbf{x}^D$ cannot be done independently from $\mathbf{x}^R$, as rigid transformations affect the object contour globally. However, we will show that advantage can be taken from the isolated estimation of $\mathbf{x}^R$ to better estimate $\mathbf{x}^D$. Before arriving to that point, first it is necessary to solve the estimation of $\mathbf{x}^R$ decoupled from $\mathbf{x}^D$, which is the topic of study in the next section.

8.1. Decoupled $\mathbf{x}^R$ estimation

The basis to estimate $\mathbf{x}^R$ without being affected by contour deformations is using with this objective the non-deformable regions of the contour to be tracked. Obviously, the existence of such regions depends on the specific object to be tracked, and requiring them to be perfectly rigid may be too restrictive for a practical use of this idea. To extend the applicability of the strategy proposed, the rigid region requirement is relaxed to use the less deformable contour zone to estimate $\mathbf{x}^R$. Obviously, the accuracy on this estimation will depend on the real rigidness of the contour part selected.

First of all, a representation of the rigidness/deformability of the points of the modeled contour is needed. As shown in Section 2.1, the space of contour deformations $\mathscr{L}(\mathbf{W}^D, \bar{\mathbf{q}})$ is established from a set of aligned contour examples, showing the different contour configurations to be modeled. From this same training set, the positional mean and covariance of contour points $(\bar{\mathbf{r}}, \mathbf{\Sigma}^{\mathbf{rr}})$ can be easily computed. The positional contour variance (computed from the diagonal values of $\mathbf{\Sigma}^{\mathbf{rr}}$) inherently maintains the rigidness or deformability of points along the modeled contour. Thus, we propose to employ it to select the contour part that can be used to estimate $\mathbf{x}^R$ confidently.

One possibility can be using $\mathbf{\Sigma}^{\mathbf{rr}}$ to set the measurement noise covariance $\mathbf{R}$, in a way that the measurements obtained in less rigid zones have less influence in the final $\mathbf{x}^R$ estimation, since the information that supply may be distorted by the local contour deformation. However, what we do in practice is using $\mathbf{\Sigma}^{\mathbf{rr}}$ to mask the non-rigid zones of the modeled shape, excluding them from the contour measurement process. That is, we only process measurement lines at i-th contour points whose positional variance $\mathbf{\Sigma}^{\mathbf{rr}}(i,i)$ is below $f\mathbf{\Sigma}_{min}$, where $\mathbf{\Sigma}_{min}$ is the minimum variance along the contour shape, and f an scaling factor (set to 500 in our experiments). In that way we reduce the computational effort required to finally estimate $p(\mathbf{x}^R|\mathbf{y}_{1:t})$.

8.2. Exploiting the $\mathbf{x}^R$ estimation: partitioned sampling

As previously stated the estimation of $\mathbf{x}^D$ requires taking $\mathbf{x}^R$ into consideration. Thanks to the proposal in the previous section, knowledge about $\mathbf{x}^R$ can be available, and we can take advantage from that. We propose to use this knowledge to define a better importance function for the PF. In principle, one may argue that this has little sense, as methods like the UPF already allow to define such an importance function, but for the whole state $\mathbf{x}$. Thus, why just taking advantage of that for $\mathbf{x}^R$? The point is that now, for the state part concerning $\mathbf{x}^R$, we are not limited to just a Gaussian approximation of the OISD: we can use an $\mathbf{x}^R$ *pre-estimation* to define it in terms of a weighted particle set. A method that exploits this idea exists, which is denoted as *partitioned sampling*.

PS is a generic term for the weight variance reduction strategy proposed in [58], based on modifying the SISR algorithm by inserting additional resampling operations in its procedure. It consists in dividing the state space into partitions, which can be propagated independently along time using their expected dynamics. Each partition is manipulated sequentially, propagating samples applying the partition dynamics followed by a *weighted resampling step*, which is the key point of this proposal. This resampling is driven by a *weighting function* that approximately evaluates the likelihood of particles from just the state partitions processed up to this point. The objective is, at each resampling step, to move particles closer to the posterior distribution to be estimated. After processing all partitions, most particles concentrate around the peaks of the posterior distribution, which improves the result achieved by the SISR algorithm. This procedure requires the following points to be fulfilled:

- the state has to be decomposable into parts $\mathbf{x} = [\mathbf{x}^{Pi}]_{i=1}^{N}$ with uncorrelated dynamics,

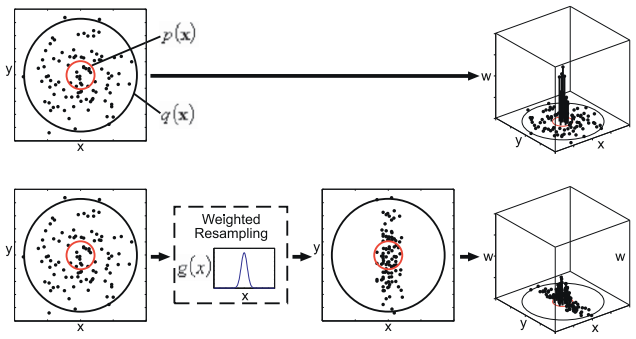

Fig. 10. Importance sampling (top) vs. partitioned sampling (bottom).

- for each state partition $\mathbf{x}^{Pi}$ a weighting function $g_{Pi()}$ is needed (continuous and strictly positive), which ideally is peaked in the same region as the likelihood of a state part composed of partitions up to $\mathbf{x}^{Pi}$ (i.e., $[\mathbf{x}^{Pj}]_{j=1}^{i}$).

In the studied contour tracking application, both conditions are accomplished by partitioning the state as $[\mathbf{x}_t^R\ \mathbf{x}_t^D]^T$. Fig. 10 compares graphically the classical importance sampling procedure, with the partitioned sampling proposal. It shows how both algorithms generate a weighted sample set of a function $p(\mathbf{x})=\mathcal{N}(\mathbf{0}_2,\sigma^2\mathbf{I}_2)$ with $\mathbf{x}=[x\ y]^T$, sampling from an importance function $q(\mathbf{x})=\mathcal{N}(\mathbf{0}_2,(4\sigma)^2\mathbf{I}_2)$. In the partitioned sampling case, a likelihood function $g(x)=\mathcal{N}(0,\sigma^2)$ is available for performing the weighted resampling. The example shows that partitioned sampling provides a sample set with minor weight variance. A drawback of this technique is that diversity is lost in the state part used in the weighted resampling, but there are methods to attenuate this problem [35,59,60].

This technique has been commonly applied to multitarget tracking applications [58,61], and to tracking applications concerning articulated models [5,62]. Here we adapt it to contour tracking based on ASMs. Before detailing how this idea is implemented for estimating $[\mathbf{x}^R\ \mathbf{x}^D]$, first the *weighted resampling* procedure is described more formally.

8.2.1. Weighted resampling method

Let $g(\mathbf{x})$ be a strictly positive, continuous function, denoted as *weighting function*. The weighted resampling of a particle set with respect to $g(\,)$ is an operation which populates the peaks of $g(\,)$ with particles, without altering the distribution actually represented by the particle set. This is carried out by a procedure derived from the importance sampling method.

Given a particle set $\{\mathbf{x}^{(i)},\tilde{w}^{(i)}\}_{i=1}^N$ approximating $p(\mathbf{x})$ as

$$\widetilde{p_N}(\mathbf{x})=\sum_{i=1}^{N}\tilde{w}^{(i)}\delta_{\mathbf{x}^{(i)}}, \tag{13}$$

first an importance function $q(\mathbf{x})$ is defined by evaluating a weighting function $g(\,)$ on the elements of this particle set. That is

$$q(\mathbf{x})=\sum_{i=1}^{N}\frac{g(\mathbf{x}^{(i)})}{\sum_{j=1}^{N}g(\mathbf{x}^{(j)})}\delta_{\mathbf{x}^{(i)}} =\sum_{i=1}^{N}\rho^{(i)}\delta_{\mathbf{x}^{(i)}}.$$

The importance sampling method states that a distribution $p(\mathbf{x})$ can be approximated from samples $\mathbf{x}'^{(i)}\sim q(\mathbf{x})$, with an associated unnormalized weight equal to $p(\mathbf{x}'^{(i)})/q(\mathbf{x}'^{(i)})$. From that result, an alternative representation of $p(\mathbf{x})$ is given by the unnormalized particle set $\{\mathbf{x}'^{(i)},w_i'\}_{i=1}^N$, where

$$\mathbf{x}'^{(i)}=\mathbf{x}^{(k_i)},$$

$$w'^{(i)}=p(\mathbf{x}'^{(k_i)})/q(\mathbf{x}'^{(k_i)}) =\tilde{w}^{(k_i)}/\rho^{(k_i)}. \tag{14}$$

Since the importance function $q(\mathbf{x})$ is defined in terms of a particle set, now we require to know which specific particle k_i from it has been assigned to $\mathbf{x}'^{(i)}$ in order to compute (14). The result of this procedure (Fig. 11) is a weighted particle set equivalent to the one in (13), but with lower weight variance according to the defined importance function $q(\mathbf{x})$.

$[\{\mathbf{x}_{0:t}'^{(i)},w_t'^{(i)}\}_{i=1}^N]=\mathrm{WR}[\{\mathbf{x}_{0:t}^{(i)},w_t^{(i)}\}_{i=1}^N,g()]$

for $i=1$ to N **do**
  $\varrho^{(i)}=g(\mathbf{x}_{0:t}^{(i)})$
end for
Compute $\rho^{(i)}=\varrho^{(i)}\Big/\sum_{j=1}^{N}\varrho^{(j)}\quad\forall i=1,\ldots,N$
Resample particle step with selected algorithm A
$[\{\mathbf{x}_{0:t}'^{(i)}\}_{i=1}^N,\{k_i\}_{i=1}^N]=\mathrm{A}[\{\mathbf{x}_{0:t}^{(i)},\rho^{(i)}\}_{i=1}^N]$
for $i=1$ to N **do**
  $w_t'^{(i)}=w_t^{(k_i)}/\rho^{(k_i)}$
end for

Fig. 11. Weighted resampling. Notice that now the resampling algorithm A must return the indexes $\{k_i\}_{i=1}^N$ of the selected particles.

$[\ \mathbf{x}_{0:t}^{(i)},w_t^{(i)}\}_{i=1}^N]=\mathrm{PS}[\{\mathbf{x}_{0:t-1}^{(i)},w_{t-1}^{(i)}\}_{i=1}^N,\mathbf{y}_{1:t}]$
{Remember that $\mathbf{x}_{0:t-1}^{(i)}\triangleq[\mathbf{x}_{0:t-1}^{R(i)}\ \mathbf{x}_{0:t-1}^{D(i)}]$}
for $i=1$ to N **do**
  Draw $\mathbf{x}_t^{R(i)}\sim p(\mathbf{x}_t^R|\mathbf{x}_{0:t-1}^{R(i)})$
  $\mathbf{x}_{0:t}^{R(i)}=[\mathbf{x}_{0:t-1}^{R(i)}\mathbf{x}_t^{R(i)}]$
end for
Weighted Resampling Step
$[\{\mathbf{x}_{0:t}^{R(i)},w_t'^{(i)}\}_{i=1}^N]=\mathrm{WR}[\{\mathbf{x}_{0:t}^{R(i)},w_{t-1}^{(i)}\}_{i=1}^N,p(\mathbf{y}_t|\mathbf{x}_t^R)]$
for $i=1$ to N **do**
  Draw $\mathbf{x}_t^{D(i)}\sim p(\mathbf{x}_t^D|\mathbf{x}_{0:t-1}^{D(i)})$
  $\mathbf{x}_{0:t}^{D(i)}=[\mathbf{x}_{0:t-1}^{D(i)}\mathbf{x}_t^{D(i)}]$
  Set $\mathbf{x}_{0:t}^{(i)}=[\mathbf{x}_{0:t}^{R(i)}\mathbf{x}_{0:t}^{D(i)}]$
  Update weight $w_t^{(i)}=w_t'^{(i)}p(\mathbf{y}_t^{(i)}|\mathbf{x}_{0:t}^{(i)})$
end for
Resampling Step
$[\{\mathbf{x}_{0:t}^{(i)},w_t^{(i)}\}_{i=1}^N]=\mathrm{R}[\{\mathbf{x}_{0:t}^{(i)},w_t^{(i)}\}_{i=1}^N]$

Fig. 12. Partitioned sampling.

8.3. Final PS algorithm

Once described the weighted resampling method, it is quite direct taking advantage of the availability of $p(\mathbf{y}_t|\mathbf{x}_t^R)$, using it to define the weighting function $\mathbf{g}(\,)$ in the algorithm of Fig. 11. Fig. 12 shows the resultant partitioned sampling method, which is based on adding the weighted resampling step in the generic SISR process of PFs.

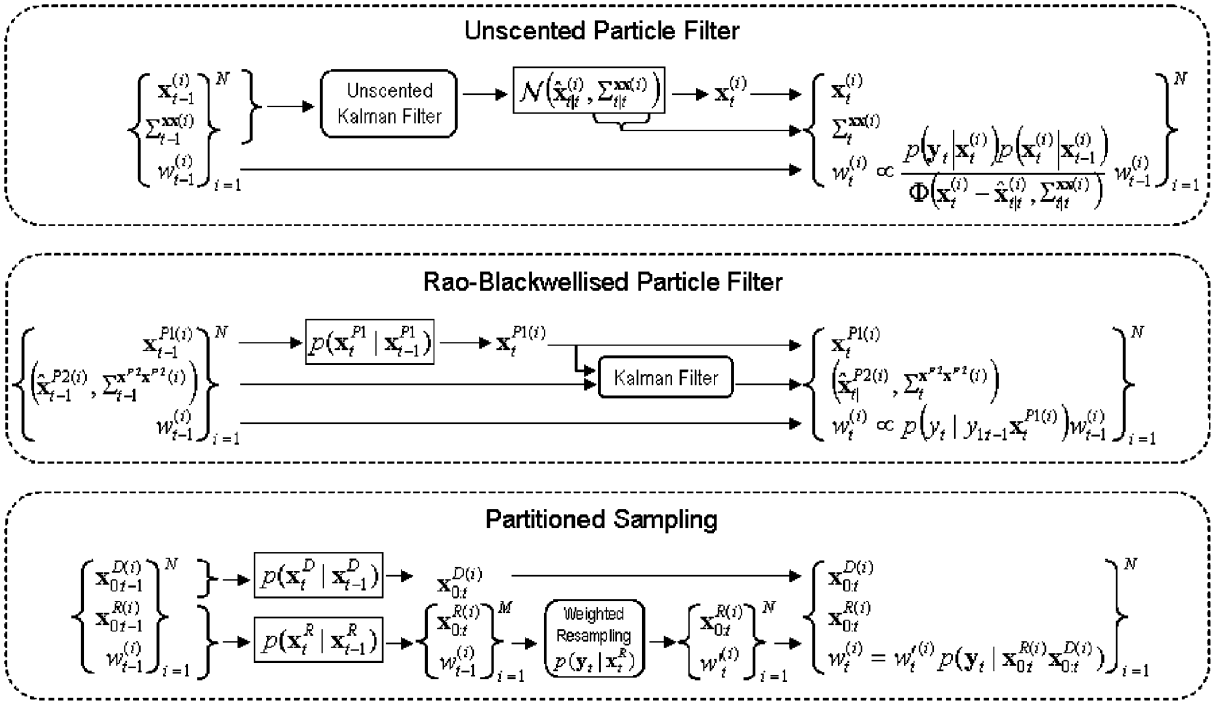

Fig. 13. Sketches of the evaluated proposals. Squared boxes remark sampled densities and rounded ones executed algorithms.

Notice that the importance function $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}\mathbf{y}_{1:t})$ inherently used in the SISR part of the algorithm corresponds to

$$p(\mathbf{x}_t|\mathbf{x}_{0:t-1}^{(i)}) = p(\mathbf{x}_t^R|\mathbf{x}_{0:t-1}^{R(i)})p(\mathbf{x}_t^D|\mathbf{x}_{0:t-1}^{D(i)});$$

that is, it is defined in terms of the dynamics of each state partition, which as required are uncorrelated.

An additional advantage of the PS algorithm is that the different resampling stages that it involves allows a more flexible distribution of the computational resources in the estimation of the different state parts. Given a partitioned state, it may happen that one of its parts has a more erratic and unpredictable behavior than the others, and consequently its dynamic model is more imprecise. This means that in every PF iteration, the region of its feasible predicted states is bigger than the one of the other parts. Thus, for a fixed number of particles, this region is less densely inspected for this part than for the others, resulting in a more inaccurate estimation of this part and, consequently, of the overall state. In these cases, a proper strategy can be adjusting the amount of particles assigned to each part according to their predictability, devoting more computational resources to the more erratic state parts. The weighted resampling step in PS (see Fig. 12) allows to implement this strategy, which was proposed in [5].

In the studied contour tracking application, rigid transformations (i.e., the whole hand movement) can change fast and unexpectedly between frames, while deformations tend to change more slowly. Thus, a number of particles $M \gg N$ should be used to inspect the feasible $\mathbf{x}^R$ states to be more robust to their lack of predictability. Then, the resultant distribution is resampled back to N samples in the weighting resampling step, to recompose the complete state and then compute its likelihood. This strategy allows, for fixed computational resources, achieving a better tracking performance. Notice that to estimate correctly $\mathbf{x}^D$, measurement lines have to provide good measurements of that. As long as these measurement lines are placed relative to the hand position (i.e., $\mathbf{x}^R$ values), the strategy of using more samples to estimate $\mathbf{x}^R$ reverts in obtaining a better $\mathbf{x}^D$ estimation, and, thus, a better overall state estimation.

8.4. Related approaches

Approaches similar to PS have been proposed in the context of tracking using multiple visual cues. In essence, the idea is partitioning the state, in order to use for each part the visual cue most informative for its estimation. In [63] this approach is used to approximate the OISD of a PF that estimates the whole state. Structurally, the idea is similar to the one of Turbo PFs (see Section 6.1), but now multiple slave filters devoted to different state parts determine a particle-based importance function for the master filter. From a different perspective, in [64] an heuristic procedure is proposed to exploit the conditional dependence between state partitions and different image cues. Several PFs are connected in serial, which estimate their corresponding state part taking advantage of the posteriors of preceding processed partitions. At the end of this process, the outputs of all PFs are combined to conform the posterior of the whole state.

Interesting variants of PS have been proposed in the context of multitarget tracking, although they are not directly applicable to contour tracking as formulated in this paper. In multitarget tracking, if targets are far apart in measurement space, each target measurement can be uniquely associated with it, and the joint posterior density of the whole state approximately factors. In this case, the multitarget tracking problem may be treated as a collection of single target problems. The independent partition PF (IPPF) [65], takes profit of this idea, and propose to apply *in parallel* a weighted resampling step for each state partition (one for each target). When the resampled partitions are joined to conform the whole state, particles with good target estimates in all partitions receive high weights and, more interestingly, particles with a low weight will have poor estimates in all partitions. This technique is not applicable when there is any measurement-to-target association ambiguity. To treat these

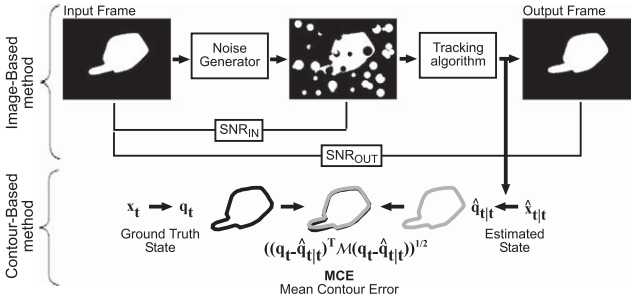

Fig. 14. Performance quantification procedure. SNR_{IN} measures the degree of distortion of the input to the tracking algorithm. SNR_{OUT} quantifies the disparity between the tracking output, and the ideal output (i.e., the undistorted input sequence). *MCE* measures the mean contour disparity between the ground truth contour and the estimated one.

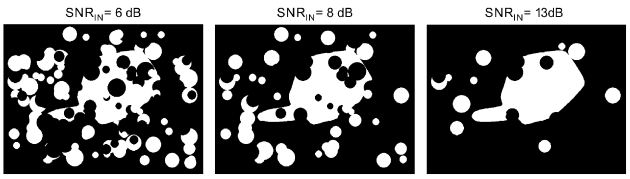

Fig. 15. Examples of noise in frames, for different situations considered in the experiments.

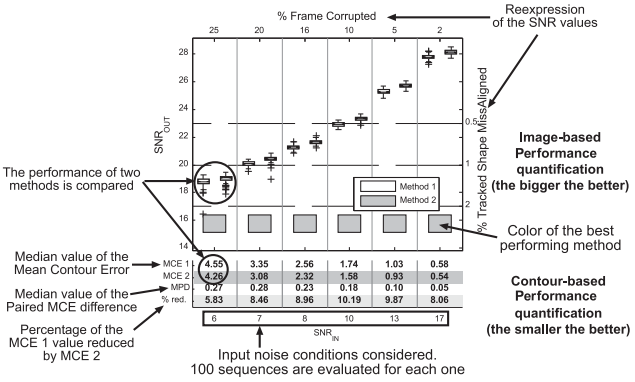

Fig. 16. Proposed representation of the evaluation results.

cases properly, in [66] is proposed the adaptive IPPF, where the state partitions are defined dynamically.

9. Experimental results

This section details the work done to objectively assess the effectiveness of proposed algorithms. Synthetic and real sequences have been processed to evaluate them quantitatively under different criteria. Examples of the performances achieved can be seen in the videos available at www.cvc.uab.es/adas/projects/contourtracking/PR/.

9.1. Synthetic sequences

With the aim of comparing proposed algorithms (see Fig. 13) under different noise situations, we have carried out the following. First, we have selected the problem of tracking a hand with pointing finger as our case study, since in this application context simultaneous local deformations and global affine changes of the tracked shape naturally appear. We have acquired a sequence showing the top view of a hand displaying a representative set of feasible configurations. Then, the outline of the hand at each frame has been annotated, generating in that way 468 examples used to train a shape and a dynamic model. With these models, we have generated different synthetic sequences in order to evaluate the performance of the different estimation methods. Generated sequences simulate the ones of a hand with a pointing finger, which has been isolated from the rest of elements in the image by means of an ideal skin segmentation process. One important advantage of processing these synthetic sequences is that tracking algorithms use the same shape model used in the sequence generation. In that way, the quality of the shape model is not a factor that can alter the performance of algorithms, as always the *perfect* model is used. Another obvious advantage is that the *ground truth* shape parameters at each frame are available, and they can be used to quantify the tracking performance.

To study the robustness of algorithms under noisy conditions, generated sequences are distorted with different artifacts prior to its processing. The tracking performance evaluation methodology is the following. Given the sequence to be analyzed, it is first distorted with random artifacts, according to a desired signal-to-noise ratio (SNR). This sequence is then processed by a tracking algorithm, obtaining the shape state at each frame. To quantify if this estimation is accurate, an *image-based* and a *contour-based* method are simultaneously applied (see Fig. 14). Essentially, the image-based method quantifies the tracking performance from the degree of overlap between the original sequence (i.e., without distortion) and a sequence generated from the estimated contour parameters. The accuracy achieved is expressed by means of an SNR value, which is higher the better the tracking performance. The contour-based method measures the average disparity between the ground truth contour shown in the sequence, and the one estimated by the tracking algorithm (i.e., the mean contour error (MCE)).

To obtain a statistical view of the performance of the algorithms, 100 noisy sequences are generated for each different level of noise considered. Fig. 15 gives some examples of different SNR_{IN} situations considered in the experiments. All trackers are evaluated on these sequences, sharing the same shape and dynamical models. To represent dynamics, a simple AR1 model has been used, while synthetic sequences have been generated using a second-order model. This has been done that way in order to evaluate methods in situations where the dynamical model roughly corresponds with the real object behavior, which commonly happens in real applications.

Two plots are generated for each proposed algorithm, comparing, respectively, their performance with respect an UKF and an standard PF. The performance measured using the image-based method is displayed using a box-plot representation, while the median of the performance measured with the contour-based method is shown in an attached table. Fig. 16 describes briefly the information provided in these performance plots.

Another criterion that we use to compare algorithms is the variance of their output (i.e., the expectation of the estimated filtering density) at each frame, in the different sequences processed for each noise level. This indicator quantifies the consistency of algorithms in providing the same estimation besides the specific sequence distortion. Hence, a high variance is obtained when an algorithm is trapped on clutter artifacts, since in each experiment they are different. However, note that a small variance does not necessarily mean a correct shape estimation, and hence this indicator just complements our performance criterion based on shape disparity.

Before comparing the performance of the different algorithms, first we need to establish the amount of particles used by them, in order to make a fair comparison. The proposed methods introduce modifications to the classical PF implementation, which require additional computational resources. From this fact, one may think of

establishing the amount of particles considered in each algorithm in order to fulfill a fixed computational cost. However, this criterion is of poor practical use, as an algorithm non-optimally implemented may be in clear disadvantage with respect to a *worse* algorithm better engineered. Moreover, the computational cost of an algorithm can vary in practice depending on the compiler used, and the characteristics of the machine available. To avoid the dependence on these factors, we propose to compare algorithms by parameterizing them in order to balance their capacity of extracting evidence from images. Hence, we set the number of particles of each algorithm, in such a way that all extract the same amount of observations per frame. In that way, what our performance evaluation methodology reflects is the ability of each strategy in taking advantage of the measurement process. Following this criterion, given that a PF with N particles performs N image measurements to evaluate their likelihood, this algorithm will be compared against an UPF with $N/2$ particles, since this one performs $N/2$ image measurements to estimate the Gaussian approximation of the OISD, and $N/2$ more to evaluate the particles likelihood. For the case of PS, $2N/3$ are considered in the weighted resampling step, while only $N/3$ conform the final particle set. Table 2 shows the relation between the size of the particle set returned by the evaluated particle-based algorithms.

Table 2
Particle set size of the evaluated particle-based algorithms.

PF	UPF	RBPF	PS
N	$N/2$	N	$N/3$

9.1.1. Performance analysis under uncorrelated clutter

Fig. 17 displays the plots of the comparative study. The analysis of these results lead us to the conclusions exposed in the following.

The UPF overcomes the UKF (i.e., has a higher SNR_{OUT}) in all the tested situations (Fig. 17(a)). With respect to a classical PF (Fig. 17(d)), it performs significantly better in all cases, except in the case of sequences with SNR $= 6$ dB. In these cases, due to the non-Gaussianity of the noise artifacts, the Gaussian approximation of the OISD is very unreliable, and results show that it is better to generate samples from the a priori assumed model of dynamics.

The RBPF outperforms PFs (Fig. 17(e)) in low-noise situations (noise $< 10\%$ pixels of frames), due to the fact that part of the state is estimated analytically. However, in these cases the UKF performs better than the RBPF (Fig. 17(b)), as this algorithm estimates analytically the whole contour state. On the other hand, in high-noise situations, the opposite behavior is observed. The RBPF does not improve the PF performance, because estimating $\mathbf{x}^{P2}$ analytically implies that the observation model is affected by Gaussian noise, and this is a wrong assumption in these cases. However, with respect to the UKF, a better performance is achieved, as estimating the state part $\mathbf{x}^{P1}$ by means of particles makes the tracker more robust to transitory misestimations of this state part. In fact, results achieved manifest the previously cited rule of thumb of estimation theory: if something can be solved analytically, do not make it by means of particles. Thus, in low-noise scenarios, estimating part of the state analytically is better than estimating the whole state using a PF, but it is worse than estimating the whole state analytically. On the other hand, in noisy situations, where the analytical solution of the contour tracking problem is bad posed, estimating part of the state by means of

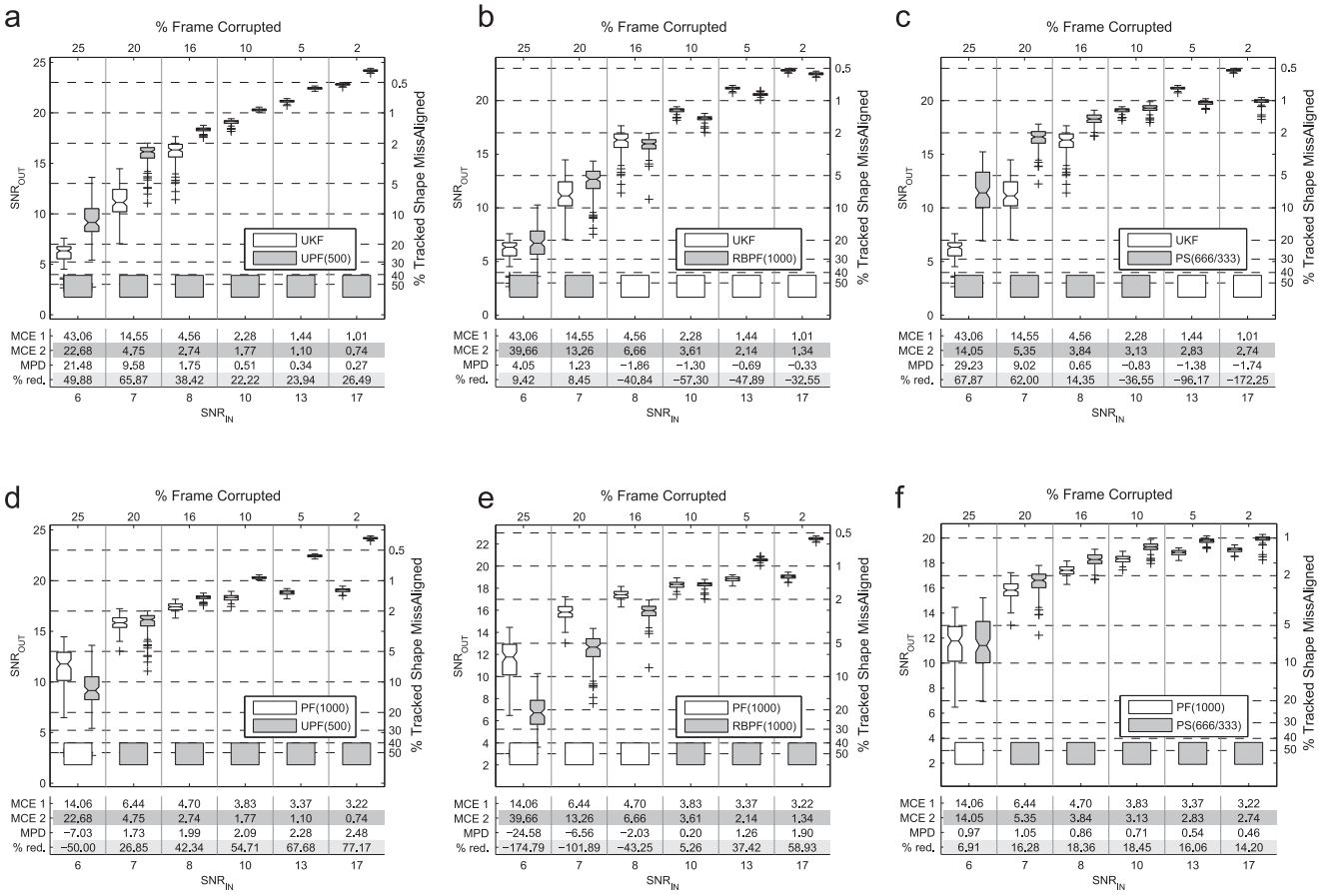

Fig. 17. Plots of the performance achieved by proposed algorithms, in comparison against the UKF (top) and the regular PF (bottom).

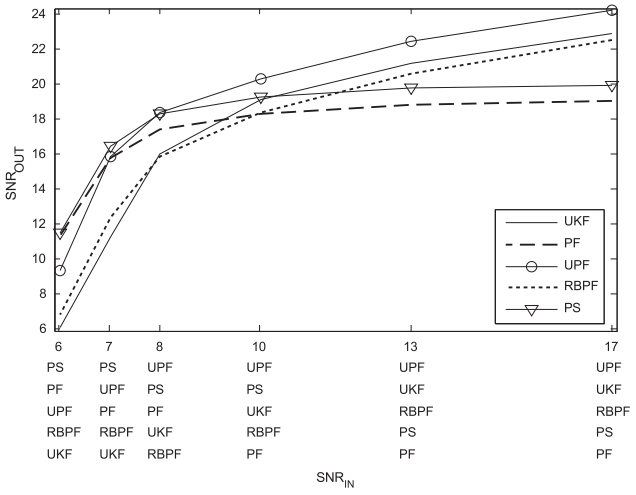

Fig. 18. Mean performance of proposed methods. For each noise situation the methods are ranked.

particles is better than estimating the whole state analytically, but worse than estimating the whole state using particles. The RBPF solution applied to contour tracking is half a way between analytic and particle-based solutions, and thus, combines the pros and cons of both methods. For this reason, it does not outperform both methods at a given noise situation. However, in specific applications where images could alternate high- and low-noise situations, it would be possible that the average performance of the RBPF could improve at the same time the one achieved by the UKF and the standard PF.

The PS algorithm has a lower performance than the UKF just in low-noise situations (Fig. 17(c)), showing a behavior similar to the one observed with the standard PF. However, the comparative study of both methods certify the superior performance of the PS with respect to a standard PF in all situations, although in the worst noise situation (i.e., $SNR_{IN} = 6$ dB) their difference is not statistically significant (Fig. 17(f)). The performance improvement achieved is due to the inner weighted resampling step in the PS, which delimits more tightly the *interesting* part of the state space to be explored. Thanks to that, more particles get closer to the ideal state parameters, and the tracking accuracy increases.

As a summary, Fig. 18 plots the mean performance of the compared algorithm at the different noise conditions considered. Clearly, the UPF is the method that provides better results in most situations, being overcame by PS only in extremely noisy sequences ($SNR_{IN} < 8$ dB). On its side, the *PS* shows its supremacy with respect to the classical PF, leveling or overcoming its performance in all situations. Finally, the RBPF applied to the contour tracking problem has turn out to be a method half-a-way between Kalman-based and particle-based filters, being never the best performing method for a given noisy situation.

The variance of the state estimations along a sequence corroborates these conclusions, ranking algorithm in the same way. Fig. 19 plots this indicator for the first shape deformation parameter, in situations of low and high noise. For the sake of clarity, results just in the first 400 frames of the sequence are plotted.

9.1.2. Performance analysis under correlated clutter

An experiment equivalent to the previous one has been performed in order to compare algorithms under correlated clutter. This kind of artifacts simulates the distortion in the shape segmentation provoked by distractors occluding or overlapping the target. We consider three situations with a noise level quantitatively lower than some cases evaluated in the previous experiments, but which are

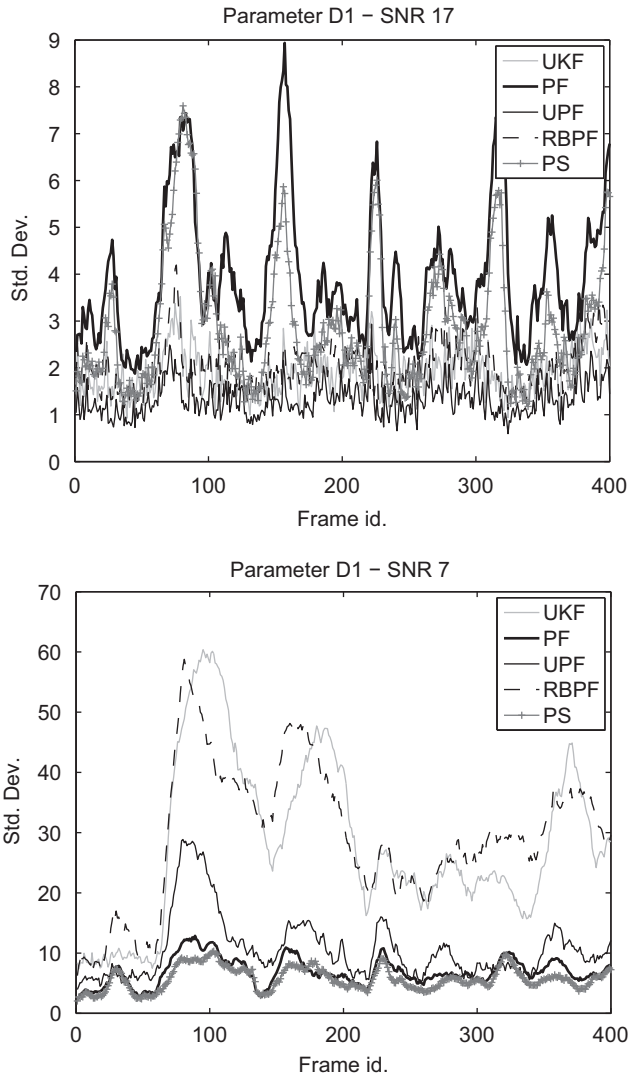

Fig. 19. Variance of the estimations of the first parameter in $\mathbf{x}^D$ under uncorrelated low (top) and high (bottom) noise conditions.

significantly more challenging in practice. The reason is that the persistence of artifacts provoke that trackers get occasionally trapped on them, losing track of the real target.

As Figs. 20 and 21 show, UPF degrades in this case very significantly, performing at a similar level than RBPF and UKF. The reason is that the Gaussian approximation of the OISD is no longer valid in these cases. This Gaussian is commonly narrow, which is in general good, since it means that particles concentrate tightly around the tracked target. However, when persistent distractors are present, the OISD is multimodal, and the UPF greedily approximate it by a Gaussian fitted just to the mode closer to the state prediction, which occasionally will correspond to clutter. The narrowness of the Gaussian provokes that particles get trapped on distractors. Correlated artifact are not a problem for PF and PS, since the importance function is defined in terms of a lax dynamical model, which inherently means using a significantly wide Gaussian. Although some particles get trapped on clutter, many of them remain close to the real target. The variance of the output of algorithms along a sequence (Fig. 22) manifests clearly the superiority of PF and PS with respect to the other methods.

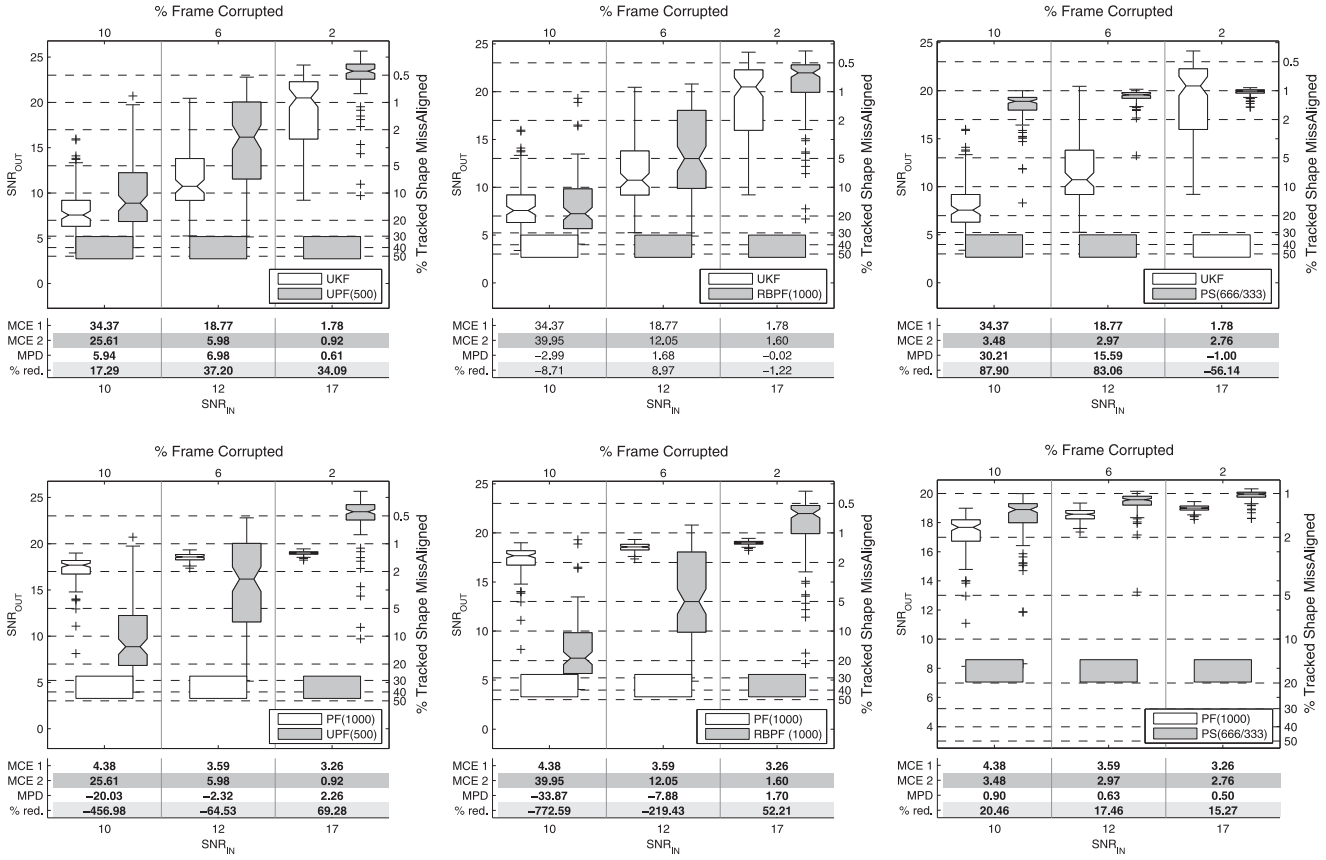

Fig. 20. Plots of the performance achieved by proposed algorithms, in comparison against the UKF (top) and the regular PF (bottom).

9.2. Real sequences

Algorithms have been also tested in real sequences concerning hand and pedestrian tracking. In the first application we verify qualitatively that algorithms perform in a real sequences coherently with the results obtained with synthetic data. In the second one we show the performance of algorithms when the tracked shape presents more severe and abrupt deformations.

9.2.1. Hand tracking

In this test we apply the same shape model used in the synthetic experiments to real sequences. A hand with a pointing finger is tracked, in sequences where the finger follows a trajectory unnoticeably marked on a paper. Sequences analyzed are preprocessed with the skin segmentation method described in [67], which isolates nicely the hand from the background (Fig. 23). However, differently than in synthetic experiments, now the shape model can just fit the tracked outline approximately. To challenge algorithms more severely, we intentionally occlude the tracked hand at some points. The performance of all compared algorithms is fair until correlated artifacts appear. In these cases, the UKF mistracks the target, the RBPF and the UPF also present problems, while PF and PS perform more robustly, which is coherent with results obtained in synthetic experiments.

We check also in these sequences how the analyzed techniques increase the number of effective particles (N_{eff}) in the density approximation. Fig. 24 show boxplots of the particle survival ratio of each algorithm prior to the resampling step (i.e., $100N_{eff}/N$). Results are provided considering all the frames in a sequence, *clean* frames, and frames where the target is occluded. In clean frames,

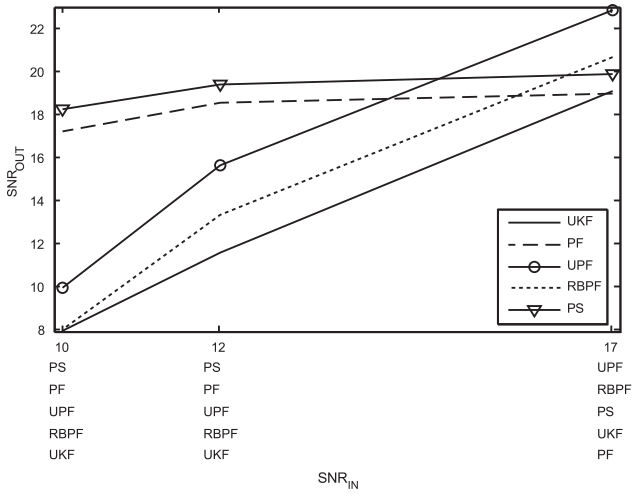

Fig. 21. Performance achieved by proposed algorithms under correlated clutter.

the goodness of tested proposals is clearly verified. However, notice the degradation of UPF and RBPF in occluded frames, due to the unfulfillment of the respective Gaussian assumption in which they are based.

Fig. 25 provides an illustrative view of the performance of the different methods under fair conditions. For each algorithm, contours corresponding to the sampling of the corresponding importance function are shown. Since particles are confined in a region

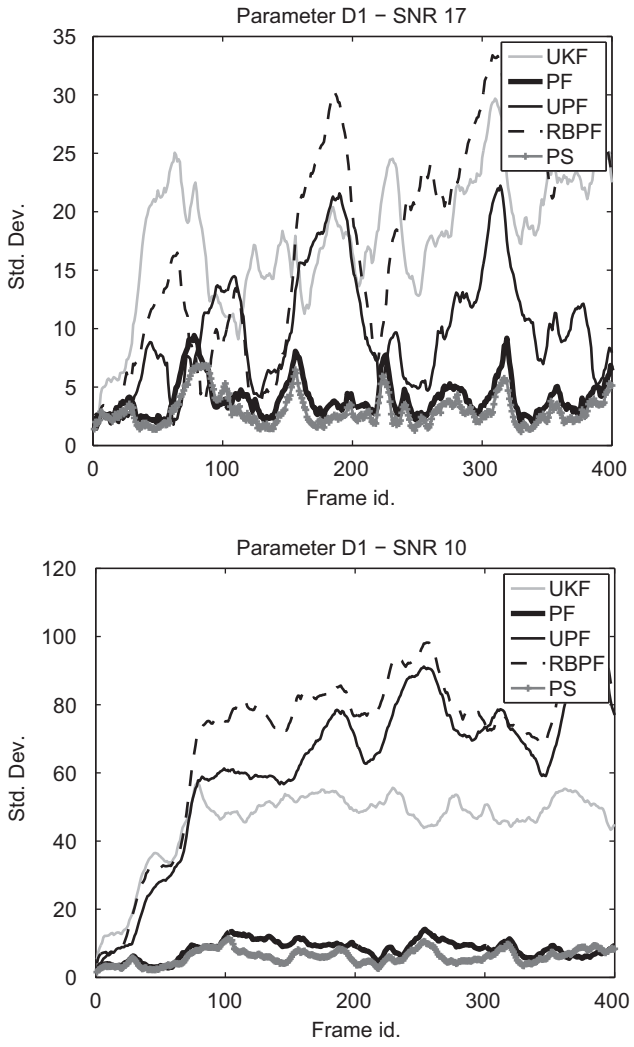

Fig. 22. Variance of the estimations of the first parameter in $\mathbf{x}^D$ under correlated low (top) and high (bottom) noise conditions.

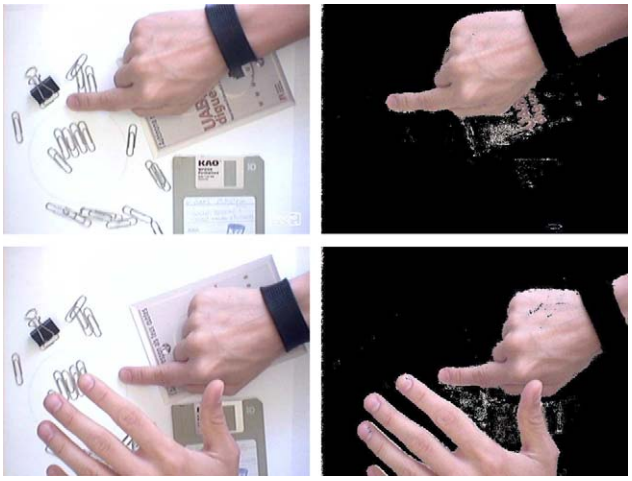

Fig. 23. Results of the skin segmentation method used.

defined using image information (UPF and PS), or inspect a lower dimensionality space (RBPF), they focus more densely on the tracked target. Since in general the likelihood of particles increases, this increases their survival ratio.

9.2.2. Pedestrian tracking

To analyze proposed algorithms in this application domain, we have trained a new shape model from 383 silhouettes manually extracted from a sequence of a pedestrian walking laterally. This has resulted in a two-dimensional model that account for severe local deformations, which occur abruptly along time. We have then applied this model in sequences representative of typical surveillance scenarios, where the global contour transformation commonly reduces to just a translation. Despite this fact, we estimate in our experiments the complete affine transformation parameters using first-order dynamics, in order to use the algorithms as formulated in the paper. Obviously this compromises the tracking results achieved (using a constant velocity model just for translation parameters would clearly improve them), but our goal in this experiment is not showing the optimal pedestrian tracking performance, but just comparing considered algorithms when tracking a highly deformable shape.

Pedestrians in the sequences are segmented using a very simple background subtraction method. A Gaussian distribution is computed for each pixel, from the statistics of their RGB values along the sequence. This conforms the background model. Then, at each frame, pixels whose RGB value is outside the 2-sigma interval of their corresponding Gaussian are labeled as foreground. By doing that, a quite rough pedestrian segmentation is obtained, consisting of blobs with irregularities due to the incorrect segmentation of some body parts (skin-colored pixels), shadows and reflections. Although this clutter could be avoided using a more sophisticated background subtraction algorithm, these conditions are interesting to test the analyzed techniques. Fig. 26 shows some results obtained in a sequence presenting uncorrelated clutter due to camera errors, and correlated artifacts due to shadows. In this case, the global best performance is achieved by PF and PS, since the poor dynamical model used does not prevent that the other algorithms get trapped on shadows. Algorithms have also been tested in a version of this sequence where shadows and other artifacts have been manually removed. Fig. 27 shows that under this conditions, the best global performance is obtained by the UPF, which corroborates the results obtained in synthetic experiments (Section 9.1).

10. Conclusions

In this paper we have proposed the novel adaptation of three well-known variance-reduction technique to the problem of particle-based visual contour tracking using ASMs: the UPF, the RBPF, and the PS algorithm. Our proposals differ from other approaches in the shape model used, which accounts for local and global contour transformations, and in a more rigorous model of the contour observation process, which leads to a more accurate interpretation of the evidence extracted from frames.

In the context of tracking the outline of a hand with a pointing finger, we have performed a comprehensive quantitative comparison of the proposed algorithms, determining the ones which lead to a better performance under different noisy conditions. Results provide a clear view of the pros and cons of the proposed techniques in general contour tracking applications. We conclude that in the presence of uncorrelated noise, except in extreme situations, the UPF is the best choice. Otherwise, the PS is the algorithm to be used. The RBPF reaches a performance between analytic and particle-based solutions, and consequently is not the best performing method for any given noise situation. When correlated artifacts are present, PS is

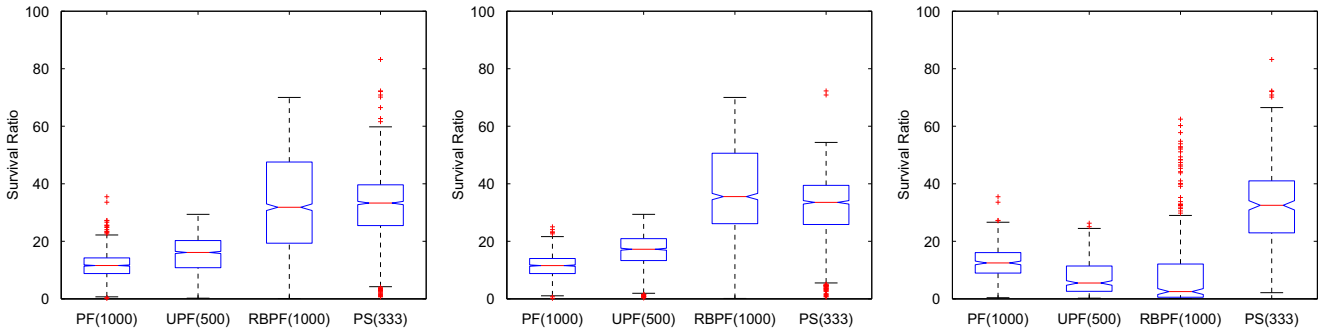

Fig. 24. Particle survival ratio of each algorithm along a whole sequence (left), in clean frames (middle) and in occluded frames (right).

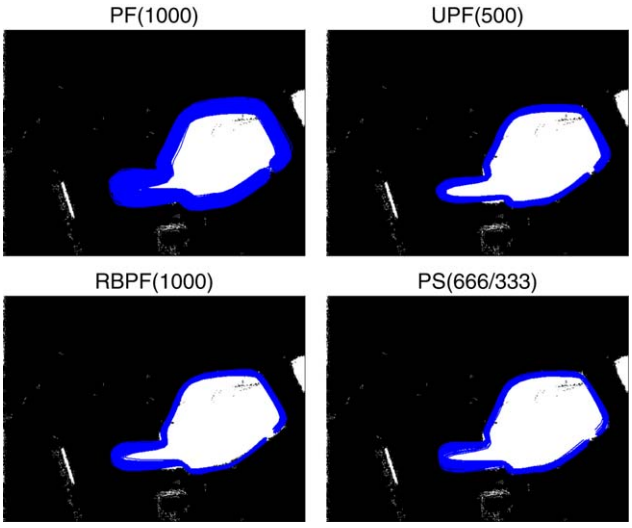

Fig. 25. Contours corresponding to samples of the importance function used in each algorithm.

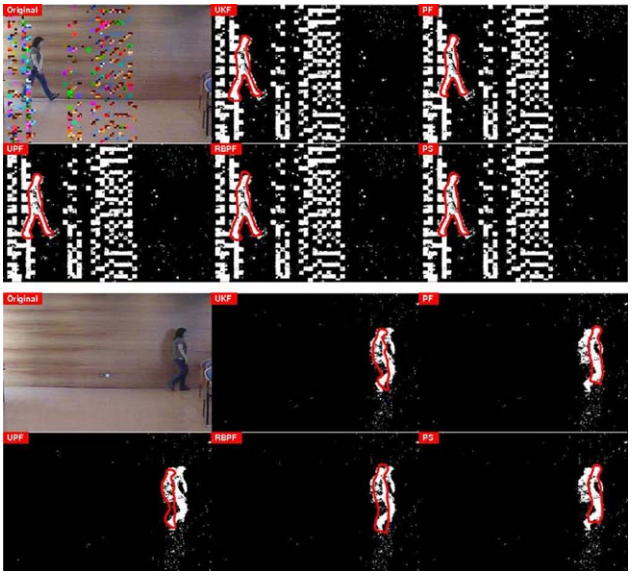

Fig. 26. Output of algorithms in frames 8 (top) and 73 (bottom) of a sequence where the pedestrian segmentation is distorted by shadows.

clearly the best performing method. In this situation, the particular Gaussian assumptions made, respectively, by UPF and RBPF penalize strongly their performance. Designing variants of their implicit strategy with the aim of improving their robustness in such cases is a natural direction for future research. Proposed algorithms have been also analyzed in real sequences, in the context of hand an pedestrian tracking. Their performance under the presence of correlated and uncorrelated clutter has corroborated the tendency observed in synthetic experiments.

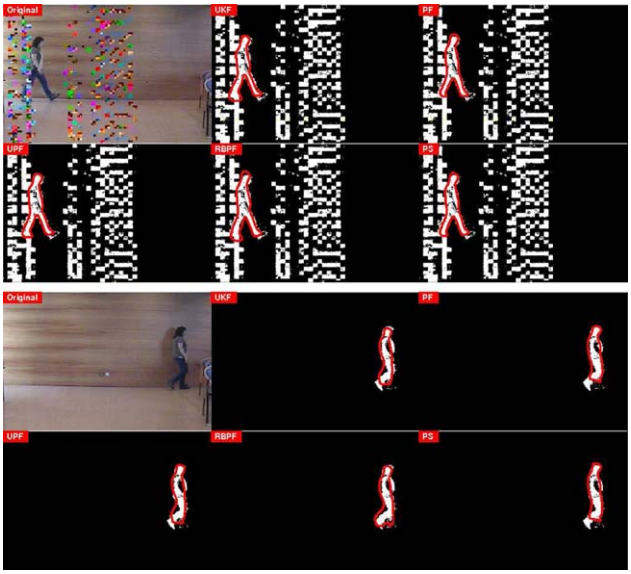

Fig. 27. Output of algorithms in frames 8 (top) and 73 (bottom) of a sequence where shadows and other artifacts have been manually removed from the pedestrian segmentation.

Acknowledgments

This work has been partially funded by Grants TRA2007–62526/AUT of the Spanish Education and Science Ministry, and Consolider Ingenio 2010: MIPRCV (CSD2007–00018).

References

[1] P. Remagnino, A. Baumberg, T. Grove, T. Tan, D. Hogg, K. Baker, A. Worrall, An integrated traffic and pedestrian model-based vision system, in: A. Clark (Ed.), Proceedings of the Eighth British Machine Vision Conference, 1997, pp. 380–389.

[2] J.-C. Tai, S.-T. Tseng, C.-P. Lin, K.-T. Song, Real-time image tracking for automatic traffic monitoring and enforcement applications, Image and Vision Computing 22 (6) (2004) 485–501.

[3] X.S. Zhou, A. Gupta, D. Comaniciu, An information fusion framework for robust shape tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (1) (2005) 115–129.

[4] J. Nascimento, J. Marques, Robust shape tracking with multiple models in ultrasound images, IEEE Transactions on Image Processing 17 (3) (2008) 392–406.

[5] J. MacCormick, M. Isard, Partitioned sampling, articulated objects, and interface-quality hand tracking, in: ECCV, vol. 2, 2000, pp. 3–19.
[6] R. Bowden, M. Sarhadi, A non-linear model of shape and motion for tracking finger spelt American sign language, Image and Vision Computing 20 (9–10) (2002) 597–607.
[7] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, International Journal of Computer Vision 1 (4) (1988) 321–331.
[9] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours, International Journal of Computer Vision 22 (1) (1997) 61–79.
[10] C. Xu, A. Yezzi, J.L. Prince, On the relationship between parametric and geometric active contours, in: 34th Asilomar Conference on Signals, Systems, and Computers, 2000, pp. 483–489.
[11] D. Freedman, T. Zhang, Active contours for tracking distributions, IEEE Transactions on Image Processing 13 (4) (2004) 518–526.
[12] N. Paragios, R. Deriche, Geodesic active regions and level set methods for motion estimation and tracking, Computer Vision and Image Understanding 97 (2005) 259–282.
[13] T. Zhang, D. Freedman, Improving performance of distribution tracking through background mismatch, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (2) (2005) 282–287.
[14] A. Doucet, S. Godsill, C. Andrieu, On sequential Monte Carlo sampling methods for Bayesian filtering, Statistics and Computing 10 (3) (2000) 197–208.
[15] A. Doucet, N. Freitas, N. Gordon, Sequential Monte Carlo Methods in Practice, Springer, New York, 2001.
[16] M.S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking, IEEE Transactions on Signal Processing 50 (2) (2002) 174–188 (see also IEEE Transactions on Acoustics, Speech, and Signal Processing).
[17] M. Isard, A. Blake, Contour tracking by stochastic propagation of conditional density, in: Proceedings of the European Conference on Computer Vision, vol. 1, Cambridge UK, 1996, pp. 343–356.
[18] M. Isard, A. Blake, CONDENSATION—conditional density propagation for visual tracking, International Journal of Computer Vision 29 (1) (1998) 5–28.
[19] A. Baumberg, Learning deformable models for tracking human motion, Ph.D. Thesis, The University of Leeds, School of Computer Studies, 1995.
[20] A. Blake, A. Isard, Active Contours, Springer, Berlin, 1998.
[21] T. Cootes, C. Taylor, Statistical models of appearance for computer vision, Technical Report, Imaging Science and Biomedical Engineering, University of Manchester, 2004.
[22] D. Cremers, Dynamical statistical shape priors for level set-based tracking, IEEE Transactions on Pattern Analysis And Machine Intelligence 28 (8) (2006) 1262–1273.
[23] Y. Rathi, N. Vaswani, A. Tannenbaum, A generic framework for tracking using particle filter with dynamic shape prior, IEEE Transactions on Image Processing 16 (5) (2007) 1370–1382.
[24] Y. Rathi, N. Vaswani, A. Tannenbaum, A. Yezzi, Tracking deforming objects using particle filtering for geometric active contours, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (8) (2007) 1470–1475.
[25] P. Tissainayagam, Visual tracking: development, performance evaluation, and motion model switching, Ph.D. Thesis, Department of Electrical and Computer Systems Engineering, Monash University, 2001.
[26] P. Tissainayagam, D. Suter, Contour tracking with automatic motion model switching, Pattern Recognition 36 (2003) 2411–2427.
[27] D. Reynard, A. Wildenberg, A. Blake, J.A. Marchant, Learning dynamics of complex motions from image sequences, in: ECCV, vol. 1, 1996, pp. 357–368.
[28] A. Wildenberg, Learning and initialisation for visual tracking, Ph.D. Thesis, University of Oxford, Robotics Research Group, Department of Engineering Science, 1998.
[29] B. North, A. Blake, Learning dynamical models using expectation-maximisation, in: ICCV, 1998, pp. 384–389.
[30] J. MacCormick, Probabilistic modelling and stochastic algorithms for visual localisation and tracking, Ph.D. Thesis, Department of Engineering Science, University of Oxford, January 2000.
[31] D. Mackay, Introduction to Monte Carlo methods, Learning in Graphical Models, MIT Press, Cambridge, MA, 1999, pp. 175–204.
[32] A. Kong, J.S. Liu, W.H. Wong, Sequential imputations and Bayesian missing data problems, Journal of the American Statistical Association 89 (425) (1994) 278–288.
[33] N.J. Gordon, D.J. Salmond, A.F.M. Smith, Novel approach to nonlinear/non-Gaussian Bayesian state estimation, IEE Proceedings—F 140 (2) (1993) 107–113.
[34] G. Kitagawa, Monte Carlo filter and smoother for non-Gaussian nonlinear state space models, Journal of Computational and Graphical Statistics 5 (1) (1996) 1–25.
[35] J.S. Liu, R. Chen, Sequential Monte Carlo methods for dynamic systems, Journal of the American Statistical Association 93 (443) (1998) 1032–1044.
[36] J. Carpenter, P. Clifford, P. Fernhead, An improved particle filter for non-linear problems, in: IEE Proceedings on Radar, Sonar and Navigation, 1999, pp. 2–7.
[37] R. Douc, O. Cappé, E. Moulines, Comparison of resampling schemes for particle filtering, in: 4th International Symposium on Image Signal Processing and Analysis, 2005, pp. 64–69.
[38] R. van der Merwe, N. de Freitas, A. Doucet, E. Wan, The unscented particle filter, in: Advances in Neural Information Processing Systems, vol. 13, 2001.
[39] Y. Rui, Y. Chen, Better proposal distributions: Object tracking using unscented particle filter, in: Conference in Computer Vision and Pattern Recognition, vol. 2, 2001, pp. 786–793.
[40] P. Li, T. Zhand, A.E. Pece, Visual contour tracking based on particle filters, Image and Vision Computing 21 (1) (2003) 111–123.
[41] L. Tierney, J.B. Kadane, Accurate approximations for posterior moments and marginal densities, Journal of the American Statistical Association 81 (393) (1986) 82–86.
[42] N. Vaswani, Particle filtering for large-dimensional state spaces with multimodal observation likelihoods, IEEE Transactions on Signal Processing 56 (10) (2008) 4583–4597.
[43] Z. Chen, T. Kirubarajan, M. Morelande, Improved particle filtering schemes for target tracking, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005. Proceedings (ICASSP '05), vol. 4, 2005, pp. iv/145–iv/148 vol. 4.
[44] E. Arnaud, E. Mémin, Partial linear gaussian models for tracking in image sequences using sequential Monte Carlo methods, International Journal of Computer Vision 74 (1) (2007) 75–102.
[45] J.-M. Odobez, D. Gatica-Perez, S.O. Ba, Embedding motion in model-based stochastic tracking, IEEE Transactions on Image Processing 15 (11) (2006) 3515–3531.
[46] A. Doucet, On sequential simulation-based methods for Bayesian filtering, Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering, 1998.
[47] C. Andrieu, J. de Freitas, A.Doucet, Sequential Bayesian estimation and model selection applied to neural networks, Technical Report CUED/F-INFENG/TR 341, Cambridge University, 1999 ⟨http://svr-www.eng.cam.ac.uk/⟩.
[48] K.P. Murphy, Bayesian map learning in dynamic environments, in: Advances in Neural Information Processing Systems (NIPS), 2000, pp. 1015–1021.
[49] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, P.-J. Nordlund, Particle filters for positioning, navigation and tracking, IEEE Transactions on Signal Processing 50 (2) (2002) 425–437.
[50] A. Doucet, N.J. Gordon, V. Krishnamurthy, Particle filters for state estimation of jump Markov linear systems, IEEE Transactions on Signal Processing 49 (3) (2001) 613–624.
[51] X. Xu, B. Li, Adaptive Rao-Blackwellized particle filter and its evaluation for tracking in surveillance, IEEE Transactions on Image Processing 16 (3) (2007) 839–849.
[52] S. Särkkä, A. Vehtari, J. Lampinen, Rao-Blackwellized particle filter for multiple target tracking, Information Fusion 8 (1) (2007) 2–15 (special issue on the Seventh International Conference on Information Fusion-Part II, Seventh International Conference on Information Fusion).
[53] Z. Khan, T. Balch, F. Dellaert, A Rao-Blackwellized particle filter for eigentracking, in: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004, vol. 2, 2004, pp. II–980–II–986.
[54] G. Schindler, F. Dellaert, A Rao–Blackwellized parts-constellation tracker, in: ICCV Workshop on Dynamical Vision; International Conference on Computer Vision, 2005, pp. 178–189.
[55] T. Schon, F. Gustafsson, P.-J. Nordlund, Marginalized particle filters for mixed linear/nonlinear state-space models, IEEE Transactions on Signal Processing 53 (7) (2005) 2279–2289.
[56] A. Kale, N. Vaswani, C. Jaynes, Particle filter with mode tracker(PF-MT) for visual tracking across illumination change, in: IEEE International Conference on Acoustics, Speech and Signal Processing, 2007. ICASSP 2007, vol. 1, 2007, pp. I-929–I-932.
[57] N. Vaswani, A. Yezzi, Y. Rathi, A. Tannenbaum, Time-varying finite dimensional basis for tracking contour deformations, in: 45th IEEE Conference on Decesion and Control, San Diego, CA, USA, December 13–15, 2006. pp. 1665–1672, doi:10.1109/CDC.2006.376972.
[58] J. MacCormick, A. Blake, A probabilistic contour discriminant for object localisation, in: ICCV, 1998, pp. 390–395.
[59] C. Andrieu, N. de Freitas, A. Doucet, Sequential MCMC for Bayesian model selection, in: IEEE Higher Order Statistics Workshop, 1999, pp. 130–134.
[60] C. Musso, N. Oudjane, F.L. Gland, Improving regularised particle filters, in: Sequential Monte Carlo Methods in Practice, Springer, New York, 2001, pp. 247–271.
[61] K. Smith, D. Gatica-Perez, Order matters: a distributed sampling method for multi-object tracking, in: British Machine Vision Conference (BMVC), No. 11, 2004, pp. 25–32.
[62] T. Yamamoto, R. Chellappa, Shape and motion driven particle filtering for human body tracking, in: ICME '03: Proceedings of the 2003 International Conference on Multimedia and Expo—Volume 3 (ICME '03), IEEE Computer Society, Washington, DC, USA, 2003, pp. 61–64.
[63] K. Branson, S. Belongie, Tracking multiple mouse contours (without too many samples), in: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005, vol. 1, 2005, pp. 1039–1046.
[64] F. Moreno-Noguer, A. Sanfeliu, D. Samaras, Dependent multiple cue integration for robust tracking, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (4) (2008) 670–685.
[65] M. Orton, W. Fitzgerald, A Bayesian approach to tracking multiple targets using sensor arrays and particle filters, IEEE Transactions on Signal Processing 50 (2) (2002) 216–223.
[66] C. Kreucher, K. Kastella, A.O. Hero III, Multitarget tracking using the joint multitarget probability density, IEEE Transactions on Aerospace and Electronic Systems 41 (4) (2005) 1396–1414.
[67] L. Sigal, S. Sclaroff, V. Athitsos, Skin color-based video segmentation under time-varying illumination, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (7) (2004) 862–877.

About the Author—DANIEL PONSA received the BSc degree in Computer Science from the Universitat Autònoma de Barcelona (UAB) in 1996, the MSc degree in Computer Vision in 1998, and the PhD in 2007. From 1996 till 2003 he worked as teaching assistant at the Computer Science Department of the UAB. He is currently a full-time researcher at the Computer Vision Center research group on advanced driver assistance systems by computer vision.

About the Author—ANTONIO M. LÓPEZ received the BSc degree in Computer Science from the Universitat Politécnica de Catalunya in 1992, the MSc degree in image processing and artificial intelligence from the Universitat Autònoma de Barcelona (UAB) in 1994 and the PhD in 2000. Since 1992 he has been giving lectures at the Computer Science Department of the UAB, where currently he is an associate professor. He is responsible of the research group on advanced driver assistance systems by computer vision in the Computer Vision Center at the UAB.