Integration of Dependent Bayesian Filters for Robust Tracking

Francesc Moreno-Noguer and Alberto Sanfeliu
Institut de Robòtica i Informàtica, UPC-CSIC
Llorens Artigas 4-6, 08028, Barcelona, Spain
Email: fmoreno, asanfeliu@iri.upc.edu

Dimitris Samaras
Computer Science Department
SUNY Stony Brook, NY 11794-4400, USA
Email: samaras@cs.sunysb.edu

***Abstract*— Robotics applications based on computer vision algorithms are highly constrained to indoor environments where conditions may be controlled. The development of robust visual algorithms is necessary for improving the capabilities of many autonomous systems in outdoor and dynamic environments. In particular, this paper proposes a tracking algorithm robust to several artifacts which may be found in real world applications, such as lighting changes, cluttered backgrounds and unexpected target movements. In order to deal with these difficulties the proposed tracking methodology integrates several Bayesian filters. Each filter estimates the state of a particular object feature which is conditionally dependent on another feature estimated by a distinct filter. This dependence provides improved representations of the target, allowing to segment it out from the background of the image. We describe the updating procedure of the Bayesian filters by a 'hypotheses generation and correction' scheme. The main difference with respect to previous approaches is that the dependence between filters is considered during the feature observation, i.e, into the 'hypotheses correction' stage, instead of considering it when generating the hypotheses. This proves to be much more effective in terms of accuracy and reliability.** [1]

I. Introduction

The development of robust tracking techniques is of enormous importance for improving the capabilities of many autonomous systems in outdoor and dynamic environments. It has been observed that the simultaneous use of redundant and complementary cues for describing a target, improves noticeably the performance of the tracking algorithms [2], [4], [5], [8], [12], [15]–[17]. Nevertheless, most of these approaches, are not robust enough and suffer from various limitations: they use to be tailored to specific applications in controlled environments; do not represent a general integration methodology which might be extrapolated to new experimental conditions; and do not take advantage of the relationship existent between different object cues. In visual tracking, we may consider this cue dependence when representing the update procedure of Bayesian filters in a hypotesis/es generation (prediction) and a hypotesis/es correction (observation) stages. During the correction of a predicted feature state, information about the state of another feature might be used. For instance, a usual method to correct the samples of a contour particle filter, is based on the ratio of the number of pixels inside the contour with object color versus the number of pixels outside the contour with background color, i.e, the contour cue depends on the color cue.

In order to overcome the limitations of previous approaches, we introduce and analyze a probabilistic framework which allows to integrate several object cues estimated by Bayesian filters (such as Kalman or particle filters). The main properties and contributions of the proposed method may be summarized as follows: 1.- It is general, in the sense that it does not restrict the total number of features to integrate. 2.- The complexity of the system does not increase noticeably when integrating additional features. 3.- Allows to represent and take advantage of cue dependence. All these properties enable the robust segmentation and tracking of objects in non-stationary environments, with abrupt changes of illumination, cluttered backgrounds and non-linear target dynamics.

The rest of the paper is organized as follows: Section II, reviews related work. In Section III, the mathematical framework is introduced. In Section IV, a comprehensible example for one dimensional cues will be explained, which will be used as a benchmark to compare the performance of our method to that of other approaches. Results in real environments and conclusions are given in Sections V and VI, respectively.

II. Related work

In fact, this is not the first work to consider multiple cue integration for tracking tasks, from a Bayesian point of view. The simplest cue integration approach, is to consider an extended state vector including the parameterization of all the cues. For instance, Isard and Blake [6] use a single state vector to integrate appearance and shape in a particle filter framework. However, as it was observed by Khan *et al.* in [7], to proceed by simply augmenting the state space is problematic since it causes an exponential expansion of the region of possible state vector configurations, and the tracking becomes extremely complex. [7] suggests using a Rao-Blackwellized particle filter, where some 'appearance-related' coefficients are integrated out of the extended state vector. This procedure reduces considerably the size of the search space, and, as a consequence, reduces the cost of tracking. Unfortunately, the generalization of this formulation to include additional features is not feasible. Generalization may be achieved by

[1]This work was supported by CICYT project DPI2004-05414 from the Spanish Ministry of Science and Technology, and by the grants from U.S Department of Justice (2004-DD-BX-1224), Department of Energy (MO-068) and National Science Foundation (ACI-0313184).

associating a different filter to each feature. Along these lines, Rasmussen and Hager [14], introduced the Joint Probability Data Association Filter (JPDAF) for tracking several targets (note that multiple target tracking can be compared to multiple cue and single target tracking). Nevertheless, this formulation does not permit to represent the dependence among several state vectors. A similar approach is presented by Leichter *et al.* [9], where Kalman and particle filters are integrated for tracking tasks, although again, assuming independence between filters, which limits the performance of the system.

The partitioned sampling technique introduced by MacCormick *et al.* [10], [11], and the related approach of Wu and Huang [18], are probably the works which are closer to the methodology presented in this paper. Partitioned sampling is specifically designed for particle filters, and allows to reduce the curse of dimensionality problem affecting these kind of Bayesian filters. The method applies the 'hypotheses generation' and 'hypotheses correction' stages, separately for different parts of the state vector. The key difference with respect to our method is that partitioned sampling considers the cue dependence during the hypotheses generation stage, whereas we consider it during the hypotheses correction. We will show how proceeding this way, tracking accuracy and reliability are significantly improved.

III. Multiple Cue Integration

Next, we will describe the mathematical background of the proposed framework. We will start by defining the cue integration process, and subsequently, we will explain how the cue dependence is considered into the observation model.

A. Integration process

In the general case, let us describe the object being tracked by a set of F features, whose configuration is specified by the state vectors $\mathbf{x}_1, \ldots, \mathbf{x}_F$, that are sequentially conditionally dependent, i.e., feature i depends on feature $i-1$. These features have an associated set of measurements $\mathbf{z}_1, \ldots, \mathbf{z}_F$, where measurement $\mathbf{z}_i$ allows to update the state vector $\mathbf{x}_i$. The conditional a posteriori Probability Density Function (PDF) $p_1 = p(\mathbf{x}_1|\mathbf{z}_1), \ldots, p_F = p(\mathbf{x}_F|\mathbf{z}_F)$ is estimated using a corresponding Bayesian filter $\mathcal{BF}_1, \ldots, \mathcal{BF}_F$. For the whole set of variables we assume that the dependence is only in one direction, i.e, $\{\mathbf{z}_k = \mathbf{z}_k(\mathbf{z}_i, \mathbf{x}_i), \mathbf{x}_k = \mathbf{x}_k(\mathbf{x}_i, \mathbf{z}_i)\} \Longleftrightarrow i < k$.

Considering this dependence relationship we can add extra terms to the PDF of each Bayesian filter. In particular, the PDF computed by $\mathcal{BF}_i$ will be $p_i = p(\mathbf{x}_i|\mathbf{x}_1, \ldots, \mathbf{x}_{i-1}, \mathbf{z}_1, \ldots, \mathbf{z}_i)$. Introducing the notation for the *cue-augmented state vector* $\mathbf{X}_{1:k} = \{\mathbf{x}_1, \ldots, \mathbf{x}_k\}$, and *cue-augmented measurement vector* $\mathbf{Z}_{1:k} = \{\mathbf{z}_1, \ldots, \mathbf{z}_k\}$, it can be easily shown that the whole a posteriori PDF may be sequentially computed, as follows:

$$\begin{aligned} P &= p(\mathbf{X}_{1:F}|\mathbf{Z}_{1:F}) \\ &= p(\mathbf{x}_1|\mathbf{Z}_1)p(\mathbf{x}_2|\mathbf{X}_1, \mathbf{Z}_{1:2}) \cdots p(\mathbf{x}_F|\mathbf{X}_{1:F-1}, \mathbf{Z}_{1:F}) \\ &= p_1 p_2 \cdots p_F \end{aligned} \quad (1)$$

Eq. 1 tells that the whole PDF can be computed sequentially, starting with $\mathcal{BF}_1$ to generate $p(\mathbf{x}_1|\mathbf{Z}_1)$ which is then used to estimate $p(\mathbf{x}_2|\mathbf{X}_1, \mathbf{Z}_{1:2})$ with $\mathcal{BF}_2$, and so on. Note that the inclusion of an extra feature $\mathbf{x}_G$ (with a measurement vector $\mathbf{z}_G$) independent from the rest, is straightforward. We just need to multiply Eq. 1 by the posterior $p(\mathbf{x}_G|\mathbf{Z}_G)$.

In the iterative performance of the method, $\mathcal{BF}_i$ receives as input at iteration t, the output PDF of its state vector $\mathbf{x}_i$ at the iteration $t-1$. Therefore, we need to write the time expanded version of the PDF for $\mathcal{BF}_i$ as

$$p_i^t = p(\mathbf{x}_i^t|\mathbf{X}_{1:i-1}^t, \mathbf{Z}_{1:i}^t, p_i^{t-1}) \quad (2)$$

The complete PDF (Eq. 1) may be then expanded as:

$$\begin{aligned} P^t &= p(\mathbf{x}_1^t, \ldots, \mathbf{x}_F^t|\mathbf{z}_1^t, \ldots, \mathbf{z}_F^t) = p(\mathbf{X}_{1:F}^t|\mathbf{Z}_{1:F}^t) \\ &= p(\mathbf{x}_1^t|\mathbf{Z}_1^t, p_1^{t-1}) \cdots p(\mathbf{x}_F^t|\mathbf{X}_{1:F-1}^t, \mathbf{Z}_{1:F}^t, p_F^{t-1}) \\ &= p_1^t p_2^t \cdots p_F^t \end{aligned} \quad (3)$$

B. Bayesian filtering

We now briefly describe how the $k-th$ Bayesian filter $\mathcal{BF}_k$, computes the posterior $p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t})$, where $\mathbf{Z}_k^{t_0:t} = \{\mathbf{z}_k^{t_0}, \ldots, \mathbf{z}_k^t\}$. Without loss of generality, here we assume that the measurements are obtained just considering observations of feature $\mathbf{x}_k$.

The formulation of the tracking problem in terms of a Bayes filter, consists in recursively update the posterior $p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t})$:

$$p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t}) \propto p(\mathbf{z}_k^t|\mathbf{x}_k^t) \int_{\mathbf{x}_k^{t-1}} p(\mathbf{x}_k^t|\mathbf{x}_k^{t-1}) p(\mathbf{x}_k^{t-1}|\mathbf{Z}_k^{t_0:t-1}) d\mathbf{x}_k^{t-1} \quad (4)$$

where $p(\mathbf{z}_k^t|\mathbf{x}_k^t)$ is the *observation (or measurement) model*, and $p(\mathbf{x}_k^t|\mathbf{x}_k^{t-1})$ represents the *dynamic model*.

Although $\mathcal{BF}_k$ may take different forms (Kalman filter, extended Kalman filter, particle filter, Rao-Blackwellised particle filter, ...), in all of the cases Eq. 4 is updated through an 'hypotheses generation - hypotheses correction' scheme. Initially, based on the *dynamic model* $p(\mathbf{x}_k^t|\mathbf{x}_k^{t-1})$ and the a posterior distribution at the previous time step $p(\mathbf{x}_k^{t-1}|\mathbf{Z}_k^{t_0:t-1})$, the state of the target is predicted according to:

$$p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t-1}) = \int_{\mathbf{x}_k^{t-1}} p(\mathbf{x}_k^t|\mathbf{x}_k^{t-1}) p(\mathbf{x}_k^{t-1}|\mathbf{Z}_k^{t_0:t-1}) d\mathbf{x}_k^{t-1} \quad (5)$$

This likelihood is subsequently corrected by the *observation model* $p(\mathbf{z}_k^t|\mathbf{x}_k^t)$:

$$p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t}) = \alpha^t p(\mathbf{z}_k^t|\mathbf{x}_k^t) p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t-1}) \quad (6)$$

where $\alpha^t = 1/p(\mathbf{Z}_k^{t_0:t})$ is a normalizing constant.

Next, we overview two different implementations of Bayes filters, namely the Kalman filter and particle filters, which are representative examples for the continuous and discrete methodologies to approximate the posterior densities, and which will be the filters used in our experiments.

1) Kalman filter: In the particular case that the observation density is assumed to be Gaussian, and the dynamics are assumed to be linear with additive Gaussian noise, equations 5 and 6 result in the Kalman filter [1]. The expressions for the densities of the dynamic model and observation model are:

$$\begin{aligned} p(\mathbf{x}_k^t|\mathbf{x}_k^{t-1}) &= \mathcal{N}(\mathbf{H}_k^t \mathbf{x}_k^{t-1}; \mathbf{\Sigma}_{k,h}^t) \\ p(\mathbf{z}_k^t|\mathbf{x}_k^t) &= \mathcal{N}(\mathbf{M}_k^t \mathbf{x}_k^t; \mathbf{\Sigma}_{k,m}^t) \end{aligned} \quad (7)$$

where $\mathbf{H}_k^t$ and $\mathbf{M}_k^t$ matrices denote the deterministic components of the models, and $\mathbf{\Sigma}_{k,h}^t$, $\mathbf{\Sigma}_{k,m}^t$ are the covariance matrices of the white and normally distributed noise assumed for the models. These expressions are plugged into the Bayes filter equations 5 and 6, which then can be analytically solved. The hypotheses generation stage provides a Gaussian likelihood, $p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t-1}) = \mathcal{N}(\mathbf{x}_{k,-}^t; \mathbf{\Sigma}_{k,-}^t)$, where:

$$\mathbf{x}_{k,-}^t = \mathbf{H}^t \mathbf{x}_k^{t-1} \qquad \mathbf{\Sigma}_{k,-}^t = \mathbf{\Sigma}_{k,h}^t + \mathbf{H}_k^t \mathbf{\Sigma}^{t-1} (\mathbf{H}_k^t)^T \quad (8)$$

Similarly, the hypotheses correction stage generates a Gaussian posterior density $p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t}) = \mathcal{N}(\mathbf{x}_k^t, \mathbf{\Sigma}_k^t)$, where:

$$\mathbf{x}_k^t = \mathbf{x}_{k,-}^t + \mathbf{K}^t[\mathbf{z}_k^t - \mathbf{M}_k^t \mathbf{x}_{k,-}^t] \qquad \mathbf{\Sigma}_k^t = [\mathbf{I} - \mathbf{K}^t \mathbf{M}_k^t] \mathbf{\Sigma}_{k,-}^t \quad (9)$$

and the matrix $\mathbf{K}^t$ is the Kalman gain.

2) Particle filter: In noisy scenes with cluttered backgrounds, observations usually have non-Gaussian multi-modal distributions, and models estimated using the Kalman filter are no longer valid. Particle filtering [3] offers an approximate solution for these cases, and approximates the posterior $p(\mathbf{x}_k^{t-1}|\mathbf{Z}_k^{t_0:t-1})$ by a set of weighted samples $\{\mathbf{s}_{kj}^{t-1}, \pi_{kj}^{t-1}\}_{j=1}^{n_k}$, where π_{kj}^{t-1} is the weight associated to particle $\mathbf{s}_{kj}^{t-1}$. The Bayes filter equation 4 is represented by:

$$p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t}) \propto p(\mathbf{z}_k^t|\mathbf{x}_k^t) \sum_{j=1}^{n_k} \pi_{kj}^{t-1} p(\mathbf{x}_k^t|\mathbf{s}_{kj}^{t-1}) \quad (10)$$

which is recursively approximated using, again, a 'hypotheses generation - hypotheses correction' strategy. Note that now, the dynamic model is represented by the distribution $p(\mathbf{x}_k^t|\mathbf{s}_{kj}^{t-1})$.

During the hypotheses generation stage, a set of n_k samples $\mathbf{s}_{kj}^t$ is drawn from the distribution $\{\mathbf{s}_{kj}^t\}_{j=1}^{n_k} \sim \sum_{j=1}^{n_k} \pi_{kj}^{t-1} p(\mathbf{x}_k^t|\mathbf{s}_{kj}^{t-1})$.

For implementation purposes, this stage is usually split into two subprocesses: initially the set $\{\mathbf{s}_{kj}^{t-1}, \pi_{kj}^{t-1}\}_{j=1}^{n_k}$, is resampled (sampling with replacement) according to the weights π_{kj}^{t-1}. We obtain the new set $\{\tilde{\mathbf{s}}_{kj}^{t-1}, \pi_{kj}^{t-1}\}_{j=1}^{n_k}$ which is subsequently propagated based on the probabilistic dynamic model to the set $\{\mathbf{s}_{kj}^t\}_{j=1}^{n_k}$.

Finally, based on the observation function $p(\mathbf{z}_k^t|\mathbf{x}_k^t)$, the set of samples $\{\mathbf{s}_{kj}^t\}_{j=1}^{n_k}$ is weighted: $\pi_{kj}^t = p(\mathbf{z}_k^t|\mathbf{x}_k^t = \mathbf{s}_{kj}^t)$.

The set $\{\mathbf{s}_{kj}^t, \pi_{kj}^t\}_{j=1}^{n_k}$ approximates the posterior distribution $p(\mathbf{x}_k^t|\mathbf{Z}_k^{t_0:t})$ at time t.

C. Introducing cue dependence into the observation model

In this subsection we will explain how cue dependence is considered within the proposed probabilistic framework.

The dependence between cues comes from the fact that in real tracking algorithms, it is common to evaluate the hypotheses generated about a specific feature, let us say $\mathbf{x}_2$, according to the state of another feature, let us say $\mathbf{x}_1$. The usually adopted solution consist in assuming the former feature $\mathbf{x}_1$ to be represented by a deterministic state vector, which at the most has been updated using some naive dynamic model. Nevertheless, we wish to consider $\mathbf{x}_1$ as a stochastic variable represented by a PDF $p(\mathbf{x}_1|\mathbf{z}_1)$, and use this PDF when evaluating the hypotheses of feature $\mathbf{x}_2$. Thus $p(\mathbf{x}_1|\mathbf{z}_1)$

Algorithm: `Observe_Dependent_Feature ( )`

Input: $p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t})$ and $p(\mathbf{x}_2^t|\mathbf{Z}_2^{t_0:t-1})$

Output: $\mathbf{z}_2^t$ (observation of feature $\mathbf{x}_2^t$)

Desc: Observe and correct the hypothesis/es $p(\mathbf{x}_2^t|\mathbf{Z}_2^{t_0:t-1})$ of feature $\mathbf{x}_2^t$ given the posterior $p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t})$ of feature $\mathbf{x}_1^t$

1) Initialize $\mathbf{x}_1^t$ posterior:

   if $\mathcal{BF}_1 \equiv \mathcal{KF}_1 \Rightarrow p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t}) = \mathcal{N}(\mathbf{x}_1^t, \mathbf{\Sigma}_1^t)$

   then $\begin{cases} \{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1} = \texttt{discretize}(\mathcal{N}(\mathbf{x}_1^t, \mathbf{\Sigma}_1^t), n_1) \\ p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t}) = \{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1} \end{cases}$

   if $\mathcal{BF}_1 \equiv \mathcal{PF}_1 \Rightarrow p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t}) = \{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1}$

   then `do_nothing()`

2) Resample $p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t})$:

   $\{\mathbf{s}_{1j}^*, \pi_{1j}^*\}_{j=1}^{n_2} = \texttt{resampling}(\{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1}, n_2)$

3) Observe $\mathbf{x}_2^t$ feature based on $\mathbf{x}_1^t$ best hypothesis/es:

   if $\mathcal{BF}_2 \equiv \mathcal{KF}_2$

   then $\begin{cases} \mathbf{x}_1^* = \sum_{j=1}^{n_2} \mathbf{s}_{1j}^* \pi_{1j}^* \\ \mathbf{z}_2^t = \mathbf{z}_2(\mathbf{x}_1^*) \end{cases}$

   if $\mathcal{BF}_2 \equiv \mathcal{PF}_2$

   then $\forall j = 1, \ldots, n_2 \quad \mathbf{do} \quad \mathbf{z}_{2j}^t = \mathbf{z}_2(\mathbf{s}_{1j}^*)$

Fig. 1. **Algorithm for observing a feature $\mathbf{x}_2$ which is dependent on a feature $\mathbf{x}_1$.** ($\mathcal{BF}$: Bayesian Filter; $\mathcal{KF}$: Kalman Filter; $\mathcal{PF}$: Particle Filter)

Function: `discretize ( )`

Input: $\mathcal{N}(\mu_x, \sigma_x)$ and n

Output: $\{s_j, \pi_j\}_{j=1}^n$

Desc: Produce n uniformly distributed samples $\{s_j, \pi_j\}_{j=1}^n$ of the Gaussian function $\mathcal{N}(\mu_x, \sigma_x)$. The sampling region is centered on μ_x and has lenght $3\sigma_x$.

$L = 3\sigma_x; \quad \Delta x = \frac{L}{n-1}$

$\forall j = 1 \ldots n \quad \mathbf{do} \begin{cases} s_j = \mu - \frac{L}{2} + j\Delta x \\ \pi_j = \mathcal{N}(\mu_x, \sigma_x)|_{x=s_j} \end{cases}$

Function: `resampling ( )`

Input: $\{\mathbf{s}_j, \pi_j\}_{j=1}^{n_1}$ and n_2

Output: $\{\mathbf{s}_k^*, \pi_k^*\}_{k=1}^{n_2}$

Desc: The set $\{\mathbf{s}_j, \pi_j\}_{j=1}^{n_1}$, is resampled (sampling with replacement) according to the weights π_j, in order to define a new set $\{\mathbf{s}_k^*, \pi_k^*\}_{k=1}^{n_2}$.

$\forall j = 1 \ldots n_1 \quad \mathbf{do} \quad c_i = \sum_{j=1}^{n_1} \pi_j$

$\forall k = 1 \ldots n_2 \quad \mathbf{do} \begin{cases} \mathbf{s}_k^* = \mathbf{s}_{\epsilon(k)} \quad \epsilon(k)\text{: min } j \text{ such that } c_j \geq \frac{k}{n_2} \\ \tilde{\pi}_k = \pi_{\epsilon(k)} \end{cases}$

$\forall k = 1 \ldots n_2 \quad \mathbf{do} \quad \pi_k^* = \frac{\tilde{\pi}_k}{\sum_{j=1}^{n_2} \tilde{\pi}_j}$

Fig. 2. **Functions `discretize()` and `weighted_resample()`, which has been adapted from [6] .**

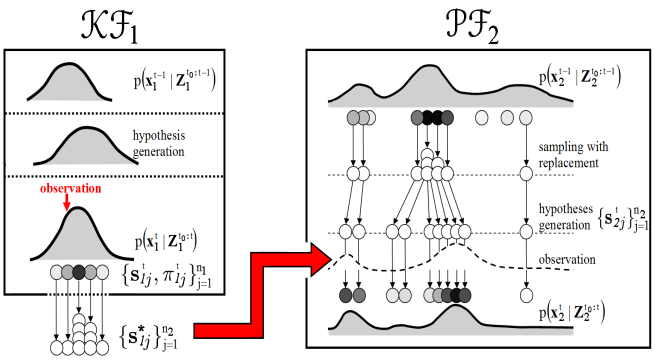


Fig. 3. **Introducing the cue dependence into the observation model**, in a case dealing with two features, one estimated by a Kalman filter and the other estimated by a particle filter. The posterior of cue $\mathbf{x}_1$, computed by $\mathcal{KF}_1$, is represented by a set of weighted samples $\{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1}$. These particles are resampled n_2 times (according to their weights), in order to obtain the set $\{\mathbf{s}_{1j}^*, \pi_{1j}^*\}_{j=1}^{n_2}$. Finally, each sample $\{\mathbf{s}_{2j}^t\}_{j=1}^{n_2}$, of cue $\mathbf{x}_2$, is weighted according to the configuration of the corresponding sample $\mathbf{s}_{1j}^*$.

is introduced into the observation model of the feature $\mathbf{x}_2$, during the 'hypotheses correction' stage of the filter.

The algorithm for introducing the state of feature $\mathbf{x}_1$ (estimated by $\mathcal{BF}_1$) into the observation model of feature $\mathbf{x}_2$ (estimated by $\mathcal{BF}_2$) is shown in Fig. 1. The key point of the integration methodology is that feature $\mathbf{x}_2$ is observed considering the best estimates of feature $\mathbf{x}_1$. The function `resampling` samples among the whole set of weighted particles approximating the posterior $p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t}) \simeq \{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1}$. The sampling is performed based on the particle weights, in such a way that the resulting subset might contain repeated copies of the more likely particles, while other elements with relatively low weights may not be considered at all. In case that $\mathcal{BF}_1$ provided a continuous PDF (such as Kalman Filter, EKF ...), the algorithm would require of an initial discretization stage, accomplished by function `discretize`. These last two functions are described in Fig. 2 [2].

Fig. 3 shows an example of how the cue dependence is handled in a case where we integrate a Kalman filter ($\mathcal{KF}_1$) and a particle filter ($\mathcal{PF}_2$). For this example, cue $\mathbf{x}_2$, estimated by $\mathcal{PF}_2$, depends on cue $\mathbf{x}_1$, estimated by $\mathcal{KF}_1$. During the observation phase of the $\mathcal{PF}_2$, the multiple hypotheses $\{\mathbf{s}_{2j}\}_{j=1}^{n_2}$, generated in the prediction stage of the filter, are weighted based on the posterior of cue $\mathbf{x}_1$, previously done by $\mathcal{KF}_1$. For this purpose, the PDF approximating $p(\mathbf{x}_1^t|\mathbf{Z}_1^{t_0:t})$ is discretized into n_1 weighted particles, $\{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1}$. Subsequently, this set is resampled n_2 times using a sampling with replacement. A set $\{\mathbf{s}_{1j}^*, \pi_{1j}^*\}_{j=1}^{n_2}$, is obtained. Finally, each sample $\mathbf{s}_{2j}^t$ of the state vector $\mathbf{x}_2$ is weighted using the configuration of cue $\mathbf{x}_1$ represented by the sample $\mathbf{s}_{1j}^*$.

Observe in Fig. 3 that the samples $\{\mathbf{s}_{1j}^t\}$ that have higher weights, have more chance to be selected several times when evaluating the hypotheses $\{\mathbf{s}_{2j}^t\}$; thus allowing to group together the more likely samples of feature $\mathbf{x}_1$ with the more likely samples of feature $\mathbf{x}_2$. Also it is important to note that not all the features need to be approximated by the same number of samples. In the example just described, $\mathbf{x}_1$ is estimated by $n_1 = 5$ samples, whereas $\mathbf{x}_2$ is estimated by $n_2 = 10$ samples. This is an important advantage of the proposed framework, especially when dealing with particle filters, since it permits to adapt the number of samples to estimate each feature, in function of its particular requirements.

To make all the mathematical foundations more clear, next we will apply the method for a simulated case, with only two one-dimensional particle filters.

IV. Dependent object features in 1D

Let us assume that we want to track a point that changes its position and color. Both features lie on a 1D space, that is, the point is moving on the horizontal axis, between the $[-1, 1]$ coordinates, and the color is also represented by a single value in the $[0, 1]$ interval. The movement of the point is simulated with a random dynamic model (centered in μ_{pos} and scaled by α_{pos}). Furthermore, we simulate an observation model, adding Gaussian noise to the simulated position :

$$\begin{aligned} \text{pos}^t &= (\text{pos}^{t-1} - \mu_{\text{pos}})\alpha_{\text{pos}} + \mathcal{N}(\mu_{\text{noise,pos}}, \sigma_{\text{noise,pos}}) \\ \text{obs\_pos}^t &= \text{pos}^t + \mathcal{N}(\mu_{\text{noise,obs\_pos}}, \sigma_{\text{noise,obs\_pos}}) \end{aligned} \quad (11)$$

Similar equations generate the models for color change (col^t) and the corresponding observation (obs_col^t).

The state of each one of the features will be estimated through particle filters. We will use $\mathcal{PF}_1$ to track the color, with $\mathbf{x}_1$ and $\mathbf{z}_1$ representing the color state vector and its measurement, and $\mathcal{PF}_2$, $\mathbf{x}_2$ and $\mathbf{z}_2$ the corresponding particle filter, state vector and measurements assigned to the position. Thus, we make the following analogies:

$$\begin{aligned} \mathcal{PF}_1: \quad & \mathbf{x}_1 = \text{col} \qquad \mathbf{z}_1 = \text{obs\_col} \\ \mathcal{PF}_2: \quad & \mathbf{x}_2 = \text{pos} \qquad \mathbf{z}_2 = \text{obs\_pos} \end{aligned}$$

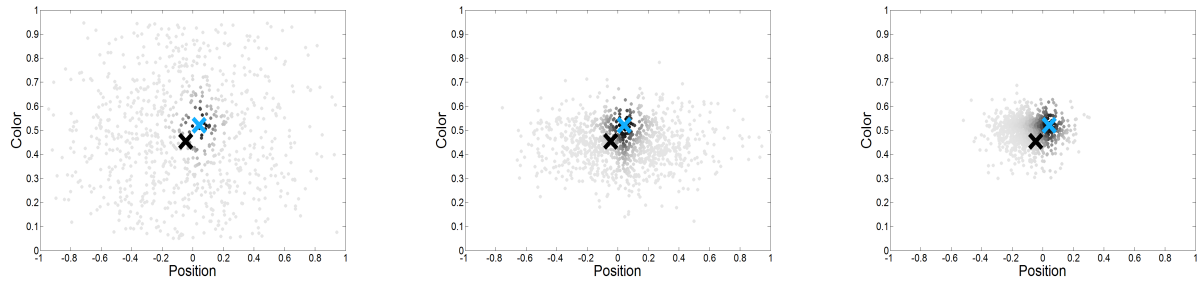
At the starting point of iteration t, $\mathcal{PF}_1$ receives at its input p_1^{t-1}, the PDF of the color at time $t-1$, approximated with n_1 weighted samples $\{\mathbf{s}_{1j}^{t-1}, \pi_{1j}^{t-1}\}_{j=1}^{n_1}$. This set is resampled and propagated according to a random dynamic model of Gaussian noise, that is, $\mathbf{s}_{1j}^t = \tilde{\mathbf{s}}_{1j}^{t-1} + \mathcal{N}(0, \sigma_{\text{dyn,col}})$, where $\tilde{\mathbf{s}}_{1j}^{t-1}$ are the resampled particles.

Each one of these propagated samples is weighted taking into account its proximity to the observed value of the color: $\pi_{1j}^t \sim e^{-(\|\mathbf{s}_{1j}^t - \text{obs\_col}^t\|)}$.

The set $\{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1}$, is the output of $\mathcal{PF}_1$ and represents an approximation to p_1^t. This PDF, jointly with p_2^{t-1}, feeds into $\mathcal{PF}_2$, which is responsible for estimating the position of the point. As in the previous particle filter, p_2^{t-1} is approximated by a set of n_2 weighted samples $\{\mathbf{s}_{2j}^{t-1}, \pi_{2j}^{t-1}\}_{j=1}^{n_2}$, which are resampled and propagated by a random Gaussian dynamic model, i.e, $\mathbf{s}_{2j}^t = \tilde{\mathbf{s}}_{2j}^{t-1} + \mathcal{N}(0, \sigma_{\text{dyn,pos}})$.

As we have previously pointed out, in real trackers, it is common to evaluate several target positions based on some appearance measure of the object, in our case, color. So we will proceed in a similar way for the weighting stage. Initially, the set $\{\mathbf{s}_{1j}^t, \pi_{1j}^t\}_{j=1}^{n_1}$ is sampled with replacement n_2 times, where the probability for each particle of being selected

[2] For ease of explanation, in the `discretize()` function we just consider the sampling of a one-dimensional Gaussian distribution. The extension to higher dimensionalities is straightforward.



(a) Condensation (b) Partitioned Sampling (c) Proposed Method

Fig. 4. **A posteriori PDF's for different particle filter based algorithms.** Comparison of the PDF's obtained by 3 algorithms when tracking a point that is randomly moving in the 'color-position' space. The results are for a particular iteration, and show how the filters approximate the true value (dark cross) based on a set of weighted particles (gray level circles). The gray level is proportional to the probability of the sample (darker gray levels indicate more likely samples). Since the true value is only ideally available, the correction of the hypotheses is done based on the observation (light cross), which we have simulated to be the true value plus Gaussian noise. In the 3 experiments, we have used the same number of particles (n=1000) and the same dynamic models. Note that the proposed approach is the method that concentrates a maximum number of samples around the true value, and therefore, it is the most reliable.

is determined by its weight π_{1j}^t. This sampling procedure generates a subset $\{\mathbf{s}_{1j}^*, \pi_{1j}^*\}_{j=1}^{n_2}$, containing the best posterior hypotheses of feature $\mathbf{x}_1$. Subsequently, each sample $\mathbf{s}_{2j}^t$, representing a position of the point in space, is associated to a corresponding color sample $\mathbf{s}_{1j}^*$, so that those color samples having larger weights (high probability) will be used more times than those having low probability. In order to simulate the weighting of the position samples taking into account the color configuration, the weight assigned to each sample $\mathbf{s}_{2j}^t$ is computed by a function $\pi_{2j}^t \sim e^{-(\|\mathbf{s}_{1j}^* - \text{obs\_col}^t\| + \|\mathbf{s}_{2j}^t - \text{obs\_pos}^t\|)}$, which simultaneously considers position and color states.

The set $\{\mathbf{s}_{2j}^t, \pi_{2j}^t\}_{j=1}^{n_2}$, represents the approximation to p_2^t. Finally, the complete a posteriori PDF of the system at time t is:

$$P^t = p(\mathbf{x}_1^t, \mathbf{x}_2^t | \mathbf{z}_1^t, \mathbf{z}_2^t) = p_1^t p_2^t$$

A. Comparison with other approaches

The simple example just presented, will be considered as a benchmark to compare the efficiency of the integration method proposed here, to that of previous approaches, specifically, with the conventional Condensation algorithm [6] assuming independent cues and the partitioned sampling algorithm [10], [11] assuming the dependence in the propagation stage.

The comparison will be performed in terms of the accuracy in the tracking (distance between the estimated position and color and the true values), and in terms of the *survival diagnostic* [11]. The survival diagnostic $\mathcal{D}$ for a particle set $\{\mathbf{s}_i, \pi_i\}_{i=1}^n$, is defined as $\mathcal{D} = \left(\sum_{i=1}^n \pi_i^2\right)^{-1}$. This random variable may be interpreted as the number of particles which would survive a resampling operation, and indicates whether the tracking performance is reliable or not. A low value of $\mathcal{D}$ means that the tracker may lose the target. For instance, if π_1=1 and π_2=...=π_n=0, then $\mathcal{D}$=1. In these circumstances only one particle might survive the resampling, and tracking would probably fail. On the other hand, if all the particles have the same weight, π_1=...=π_n=$1/n$ results in that $\mathcal{D}$=n. This indicates that all the n particles would survive an ideal resampling, and the tracking would not get lost. Made this clarification, we proceed to study the performance of different algorithms in the tracking problem proposed in this section.

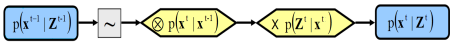
Initially, the problem has been examined by the conventional Condensation algorithm, assuming independent cues. $\mathbf{x}_1$ and $\mathbf{x}_2$ are represented into the same state vector, and the hypotheses generation and correction stages are applied simultaneously to both features. Since the dynamic model of a specific feature has no clue about the state of the other feature, particle samples are spread on a wide area of the state space and only a few particles will be located in the neighborhood of the true state. Fig. 4a shows the a posteriori PDF obtained in one iteration of the algorithm. The dots represent the different samples (in the 'color-position' configuration space), and the crosses are the true value (dark cross) and observed value (light cross). The particles gray level is proportional to their likelihood (darker levels are more likely samples). Note that only a small set of particles have a large weight. Consequently, the survival diagnostic for this approach will have low values.

Better results may be obtained using the partitioned sampling. In this case, the dynamics and measurements are not applied simultaneously, but are partitioned into two components. First, the dynamics are applied in the $\mathbf{x}_1$ direction, and subsequently the particles are rearranged so that they concentrate around the color observation (by a process called *weighted resampling* [10]). This arrangement enhances the estimation by concentrating more particles around the true state. Note in Fig. 4b this effect on the posterior PDF. Although particles are spread in the $\mathbf{x}_2$ direction, their variability along the $\mathbf{x}_1$ direction is highly reduced. As a result, the number of samples having a large weight is considerably bigger than when using the conventional Condensation. However, it is important to note that in the partitioned sampling, particles are propagated in the direction $\mathbf{x}_2$ according to the likelihood of the samples of feature $\mathbf{x}_1$. Thus, best hypotheses of feature $\mathbf{x}_1$ have more chances to be propagated in the direction $\mathbf{x}_2$. Although this approach outperforms the conventional Condensation, it still has a limitation, in that the best samples of feature $\mathbf{x}_1$ do not need to be the best samples of feature $\mathbf{x}_2$. Therefore, the common association of the best samples of feature $\mathbf{x}_1$ with the best samples of feature $\mathbf{x}_2$, is not guaranteed.

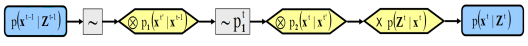
This is improved in the integration algorithm proposed in this paper. The key difference with respect to the previous

approaches, is that we consider a different state vector for each cue, and the hypotheses generation and correction stages are also applied separately. In particular, the sample propagation for cue $\mathbf{x}_i$, is performed according to the samples resampling its own PDF in the previous time step $p(\mathbf{x}_i^{t-1}|\mathbf{Z}_i^{t-1})$, and not according to the samples that better approximate another cue, avoiding the mentioned problem suffered by partitioned sampling. Fig. 4c, shows that proceeding this way the samples are much more concentrated around the true value than in previous approaches, providing larger survival diagnostic rates. Furthermore, while partitioned sampling considers the cue dependence during the hypotheses generation stage, we consider it in the correction phase, where the posterior of a specific cue is used to weigh the samples of another cue. This allows to update all the cues in a same iteration.

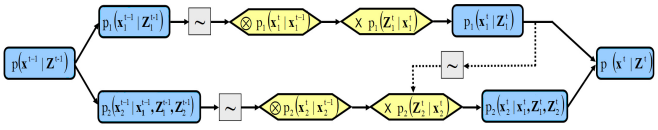
Considering the representation used in [11] to describe particle filter processes (explained by convolution and multiplication of PDF's) in Fig. 5 we describe one time step of the conventional Condensation algorithm, the partitioned sampling and the proposed algorithm. These diagrams clearly reflect the differences between all the methods:



(a) Conventional Condensation ($\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2]$)



(b) Partitioned Sampling ($\mathbf{x} = [\mathbf{x}_1, \mathbf{x}_2]$)

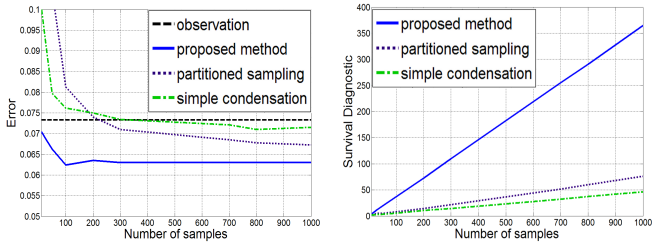


(c) Proposed Method

Fig. 5. **Whole process diagrams of the Condensation, partitioned sampling and the proposed algorithm.** The symbols are adapted from [11]: '$\sim$' is the resampling operation, '$\sim p_1^t$' denotes a weighted resampling with respect to the importance function p_1^t, '$*$' indicates a convolution with the dynamics and '$\times$' is the multiplication by the observation density.

The plots of Fig. 6 show the tracking results obtained for all the algorithms. Fig. 6a compares the methods in terms of the tracking error, where the error is computed as the distance between the filter estimate and the true value. For instance, given a posterior approximated by the set $\{\mathbf{s}_j, \pi_j\}_{j=1}^n$, and the true state of the tracked point given by $\mathbf{x}_{true}$, the value of the error is $\mathcal{E}(n) = \|E(\mathbf{x}) - \mathbf{x}_{true}\|$, where $E(\mathbf{x})$ is the expected value approximated by the filter, i.e, $E(\mathbf{x}) = \sum_{j=1}^n \mathbf{s}_j \pi_j$, and $\|\cdot\|$ refers to the Euclidean norm. Observe that the error produced using the method proposed in this paper is clearly smaller than the produced by the other algorithms.

When analyzing the survival diagnostic for the same experiments, we may reach similar conclusions. From Fig. 6b it can be seen that the largest survival rates, and hence the most reliable tracking results, are obtained when using the integration technique presented here.



(a) (b)

Fig. 6. **Tracking results obtained for the conventional Condensation, partitioned sampling and the proposed method.** Analysis of the three algorithms when are applied to the tracking example explained in this section, which was a 20 iterations sequence. The analysis is done in terms of the error in the tracking (a) and in terms of the survival rate (b). In both cases the experiments have been realized for different number of samples, and for each specific number of samples, 25 repetitions of the simulation have been done. The results we show, correspond to the mean of these 25 repetitions, of 20 iterations each. Observe that the results agree with the a posteriori PDF's plotted in Fig 4, and the proposed method clearly outperforms both the conventional Condensation and the partitioned sampling algorithms.

Just a final remark, concerning to the number of particles necessary to achieve a desired level of performance. It is well known that the *curse of dimensionality* is one of the main problems affecting particle filters, that is, when the dimensionality of the state space increases, the number of required samples increases exponentially [7], [10], [18]. Intuitively, the number of samples is proportional to the volume of the search space. For instance, if a 1D space is sampled by n particles, the same sampling density in a 2D space will require n^2 particles, and so on. Nevertheless, in the proposed method, the high dimensional state vectors are separated into various small state vectors and the sampling is particularized for each low dimensional configuration space. The final number of required particles corresponds to the sum of the particles used in each of these low dimensional spaces. For example, if a 2D state vector can be separated into two 1D state vectors, the number of samples may be reduced from n^2 (required in the 2D configuration space) to $2n$ (required in the two 1D spaces). Furthermore, as we have previously pointed out, the number of samples may be adapted for the particular requirements of each component of the whole state vector.

V. Results in Real Environments

We have used the proposed approach to track several targets in natural and unconstrained environments. For all the experiments we will show in this section, four different object cues have been used: $\mathbf{x}_1$ = object bounding box, estimated by a Kalman filter $\mathcal{KF}_1$; $\mathbf{x}_2$ = object dependent colorspace, estimated by a particle filter $\mathcal{PF}_2$; $\mathbf{x}_3$ = color distribution, estimated by $\mathcal{PF}_3$; and finally $\mathbf{x}_4$ = object contour, estimated by $\mathcal{PF}_4$. The observation functions are designed in such a way that a sequential conditional relation is satisfied, i.e, observation function $\mathbf{z}_i$ uses the state of feature $\mathbf{x}_{i-1}$. For a detailed description of such features and their corresponding observation functions, we refer the reader to [13].

Figures 7, 8 and 9 show several result frames of various tracking experiments, in challenging environments with dif-

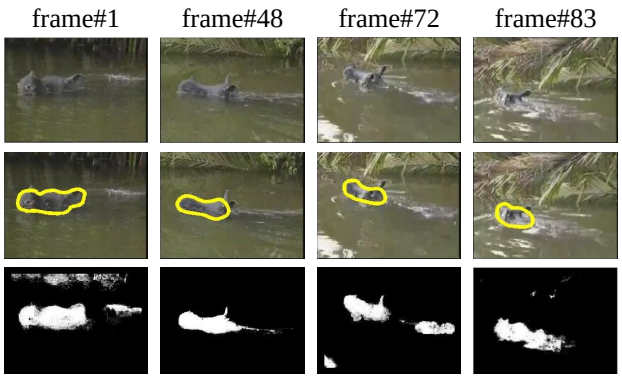


Fig. 7. **Experiment 1: Tracking an hippopotamus into the water.**

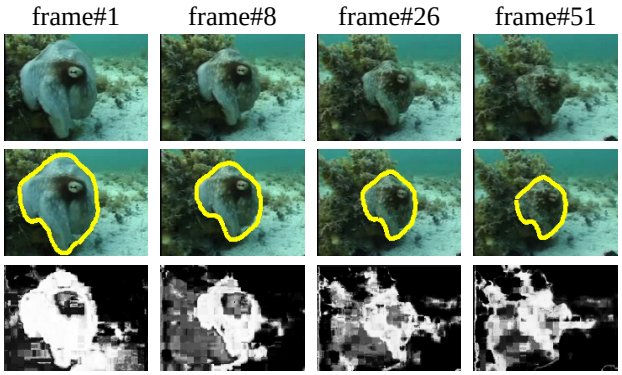


Fig. 8. **Experiment 2: Tracking a camouflaging octopus.**

ferent artifacts, such as cluttered backgrounds, camouflaging objects, gradual and abrupt illumination changes, and unexpected target movements. In all the experiments, the first row depicts the original frames, the second row are the tracking results, and the third row represents the a posteriori PDF map of the color module. Fig. 7 shows the tracking of an hippopotamus in the water. In spite of the appearance changes of the target produced by water specularities, the proposed algorithm succeeds in the tracking. More complicated is the tracking of the camouflaging octopus of Fig. 8, since its aspect changes gradually until confusing with the background. Observe that although the a posteriori PDF map of the color module provides just a rough estimate of the target position, this is subsequently corrected by the contour feature. The combination of both types of features (appearance and geometry) provides robustness to the tracking. In the last experiment we show the tracking of a moving leaf, in a sequence with an abrupt illumination change (note this change between the consecutive frames 95 and 96). Furthermore, the movement of the leaf is unpredictable, continuously varying its direction and acceleration. Under these circumstances, tracking can be achieved because of the use of particle filters for characterizing both the color and the contour of the leaf.

VI. Conclusions

In this paper we have proposed and analyzed a robust methodology to integrate several Bayesian filters for tracking tasks, where each filter is used to estimate the state of a different object feature. This framework is particularly interesting for performing tracking in outdoor scenarios, where abrupt illumination changes, cluttered backgrounds and unexpected target dynamics might occur. The key point of our technique is the consideration of the cue dependence when observing each particular feature in order to correct its hypothesized state, as opposed to previous approaches, which used to consider the cue dependence during the hypotheses generation stage. Our method proves to be much more effective in terms of accuracy and reliability, as we show through a set of synthetic and real tracking experiments.

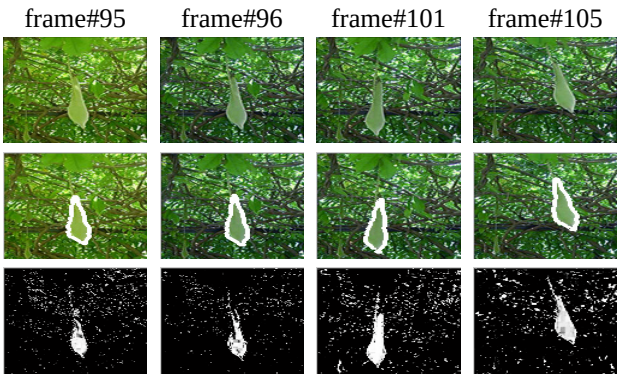


Fig. 9. **Experiment 3: Tracking a moving leaf.**

References

[1] Y.Bar-Shalom, X.R.Li, T.Kirubarajan, "Estimation with applications to tracking and navigation", *John Wiley & Sons, New York*, 2001.
[2] S.Birchfield, "Elliptical head tracking using intensity gradients and color histograms", *CVPR*, pp.232-237, 1998.
[3] A.Doucet, N.de Freitas, N.Gordon, editors, "Sequential Monte Carlo in practice", *Springer-Verlag, New York*, 2001.
[4] G.Hager, P.Belhumeur, "Efficient region tracking with parametric models of geometry and illumination", *PAMI*, Vol.20(10), pp.1125-1139, 1998.
[5] E.Hayman, J.O.Eklundh, "Probabilistic and voting approaches to cue integration for figure-ground segmentation", *ECCV*, pp.469-486, 2002.
[6] M.Isard, A.Blake, "Condensation-conditional density propagation for visual tracking", *IJCV*, Vol.29(1), pp.5-28, 1998.
[7] Z.Khan, T.Balch, F.Dellaert, "A rao-blackwellized particle filter for eigentracking", *CVPR*, pp.980-986, 2004.
[8] S.Khan, M.Shah, "Object based segmentation of video using color, motion, and spatial information", *CVPR*, Vol.2, pp.746-751, 2001.
[9] I.Leichter, M.Lindenbaum, E.Rivlin, "A probabilistic framework for combining tracking algorithms", *CVPR*, 2004.
[10] J.MackCormick, A.Blake, "Probabilistic exclusion and partitioned sampling for multiple object tracking", *IJCV*, Vol.39(1), pp.57-71, 2000.
[11] J.MacCormick, M.Isard,"Partitioned sampling, articulated objects, and interface-quality hand tracking", *ECCV*, Vol.2, pp.3-19, 2000.
[12] J.Malik, S.Belongie, J.Shi, T.Leung, "Textons, contours and regions: Cue integration in image segmentation", *ICCV*, pp.918-925, 1999.
[13] F.Moreno-Noguer, A.Sanfeliu, D.Samaras "Integration of conditionally dependent object features for robust figure/background segmentation", *ICCV*, pp.1713-1720, 2005.
[14] C.Rasmussen, G.D.Hager, "Probabilistic data association methods for tracking complex visual objects", *PAMI*, Vol.23(6), pp.560-576, 2001.
[15] P.Torr, R.Szelinski, P.Anandan, "An integrated bayesian approach to layer extraction from image sequences", *PAMI*, 2001.
[16] K.Toyama, E.Horvitz, "Bayesian modality fusion: Probabilistic integration of multiple vision cues for head tracking", *ACCV*, 2000.
[17] J.Triesch, C.von der Malsburg, "Democratic integration: self-organized integration of adaptive cues", *Neural Computation*, Vol.13(9), pp.2049-2074, 2001.
[18] Y.Wu, T.S.Huang, "Robust visual tracking by integrating multiple cues based on co-inference learning", *IJCV*, Vol.58(1), pp.55-71,2004.