A self-adaptive likelihood function for tracking with particle filter

Séverine Dubuisson[1], Myriam Robert-Seidowsky[2] and Jonathan Fabrizio[2]
[1] *CNRS, UMR 7222, ISIR, F-75005, Paris, France*
[2]*LRDE-EPITA, 14-16, rue Voltaire, F-94276, Le Kremlin Bicêtre, France*
severine.dubuisson@isir.upmc.fr {*myriam,jonathan*}*@lrde.epita.fr*

Keywords: visual tracking, particle filter, likelihood function, correction step.

Abstract: The particle filter is known to be efficient for visual tracking. However, its parameters are empirically fixed, depending on the target application, the video sequences and the context. In this paper, we introduce a new algorithm which automatically adjusts online two majors of them: the correction and the propagation parameters. Our purpose is to determine, for each frame of a video, the optimal value of the correction parameter and to adjust the propagation one to improve the tracking performance. On one hand, our experimental results show that the common settings of particle filter are sub-optimal. On another hand, we prove that our approach achieves a lower tracking error without needing to tune these parameters. Our adaptive method allows to track objects in complex conditions (illumination changes, cluttered background, *etc.*) without adding any computational cost compared to the common usage with fixed parameters.

1 INTRODUCTION

Visual tracking is an important task in computer vision with a lot of applications in our daily life, such as monitoring human-computer interaction for example. Among all existing trackers, those relying on the particle filter have become very popular thanks to their simple algorithmic scheme and efficiency. However, one of the main drawback of the particle filter is that it requires to fix two parameters (see Section 2) that drastically influence the tracking performances. The first one (σ parameter) is used within the propagation step, and fixes how far the particles should be diffused from the current estimated state. The second one (α parameter) is related to the correction step and influences how much the likelihood function is peaked. Although they are essential, they are arbitrarily fixed in most implementations. In this paper, we propose an approach that can automatically determine their optimal value depending on the temporal context. We have proven the efficiency of our method on several challenging video sequences. Moreover, there is no additional computational cost compared to a version with fixed parameters.

Only few works discuss the correction parameter α and the propagation parameter σ settings or their adaptation over time. The authors in (Fontmarty et al., 2009) study the impact α and σ values for human tracking. In their case, they use four specific likelihoods related to the silhouette of the tracked object. They attempt to tune the α value with respect to an empirical study on their own data. By observing the behavior of tracking in different configurations, they determine its optimal interval of values for each likelihood. Although this is a very interesting work, we can not generalize its conclusion, because the study was made for a specific context with dedicated likelihoods. The work proposed in (Lichtenauer et al., 2004) is also an empirical study to determine an optimal value for α. Here, different distances for comparing particles and model (gradient direction and Chamfer matching) are considered for single and multiple object tracking cases. Experiments, as for previous work, permit to determine the optimal range of values for α. Here again, the choice of the α value highly depends on the chosen distance. In (Brasnett and Mihaylova, 2007), a problem of multi-cue tracking (color, texture and edges) is addressed and an adaptive scheme for the computation of the correction parameter is proposed. To the best of our knowledge, this is the only work that attempts to provide a general definition of α value. Authors then propose the following heuristic: $\alpha = \frac{1}{\sqrt{2}d_{min}}$. Even if their purpose is to have a particle set containing as few particles with low weights as possible, their heuristic is not well justified. In particular, the adaptation of the α value to the minimal distance d_{min} of the particles set permits to avoid particles to be located on the tail of the like-

lihood distribution. To our own, this is however important for some particles to have high weights, but also for some of them to have low weights to preserve a diversity within the particle set. Note that a similar work was proposed in (Ng and Delp, 2009). Except for the latter article, previous solutions rely on empiric tests and thus, are dedicated to specific tracking contexts. In this paper, we propose an algorithm that can adapt α and σ values over time depending on the context. It uses a single and simple criterion based on a potential survival rate and the maximum weight of the particles set. The paper is organized as follows. Section 2 presents the particle filter and its current implementation. It also explains the role of the propagation and correction parameters. Section 3 details our approach to online automatically adapt α value and correct σ if necessary. Section 4 gives experimental results *via* both qualitative and quantitative evaluations. Finally, concluding remarks are given in Section 5.

2 PARTICLE FILTER

Theoretical framework. The goal of particle filter's framework (Gordon et al., 1993) is to estimate a state sequence $\{\mathbf{x}_t\}_{t=1,\dots,T}$, whose evolution is given from a set of observations $\{\mathbf{y}_t\}_{t=1,\dots,T}$. From a probabilistic point of view, it amounts to estimate for any t, $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. This can be computed by iteratively using Eq. (1) and (2), which are respectively referred to as a prediction step and a correction step.

$$p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1} \quad (1)$$

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \propto p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) \quad (2)$$

In this case, $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is the transition and $p(\mathbf{y}_t|\mathbf{x}_t)$ the likelihood. Particle filter aims at approximating the above distributions using weighted samples $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}$ of N possible realizations of the state $\mathbf{x}_t^{(i)}$ called *particles*.

The global algorithm of particle filter between times steps $t-1$ and t is summarized below.

1. Represent filtering density at $t-1$ $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ by a set of weighted particles (*i.e.* samples) $\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\}$, $i = 1,\dots,N$
2. Explore the state space by propagating samples using a proposal function so that $\mathbf{x}_t^{(i)} \sim q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_{1:t})$.
3. Correct samples by computing particle's weight so that:

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{0:t-1}^{(i)}, \mathbf{y}_{1:t})} \quad (3)$$

with $\sum_{i=1}^{N} w_t^{(i)} = 1$

4. Estimate the posterior density at t by the weighted samples $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}$, $i = 1,\dots,N$, *i.e.*

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta_{\mathbf{x}_t^{(i)}}(\mathbf{x}_t)$$

where $\delta_{\mathbf{x}_t^{(i)}}$ are Dirac masses centered on particles.

5. Resample (if necessary)

Model for implementation. Two densities are fundamental for particle filter: the propagation and the likelihood ones.

The first one permits to get the prior of the estimation, and rely on the proposal density $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{y}_{1:t})$. The most common choice for this proposal is the transition density $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. The resulting recursive filter is also called Bootstrap filter, or CONDENSATION (Gordon et al., 1993). Because the motion of the tracked object is unknown, the transition function is often modeled by a Gaussian random walk centered around the current estimation $\mathbf{x}_{t-1}$. We then get $p(\mathbf{x}_t|\mathbf{x}_{t-1}) \sim \mathcal{G}(\mathbf{x}_{t-1}, \Sigma)$, where $\mathcal{G}$ is a Gaussian with mean $\mathbf{x}_{t-1}$ and covariance matrix Σ. The variances σ (in the diagonal of Σ) fix how far the particles are propagated from $\mathbf{x}_{t-1}$. The higher these variances, the further particles are propagated.

When using transition function as proposal, Eq. 3 becomes $w_t^{(i)} \propto w_{t-1}^{(i)} p(\mathbf{y}_t|\mathbf{x}_t^{(i)})$. In such a case, particles' weights are proportional to the likelihood, and if we only consider the current observation, we get $w_t^{(i)} \propto p(\mathbf{y}_t|\mathbf{x}_t^{(i)})$. According to the common modeling of the likelihood, the second fundamental density is $p(\mathbf{y}_t|\mathbf{x}_t^{(i)}) \propto e^{-\alpha d^2}$, where e is the exponential function, d is a similarity measure ($d \in [0,1]$) that indicates the proximity between particle $\mathbf{x}_t^{(i)}$ and the model of the tracked object. Usually, the tracked object is modeled by the histogram of its surrounding region, and d is the Bhattacharya distance (Bhattacharyya, 1943). Parameter α influences how peaked is the likelihood function: if α value is high (resp. small), then the likelihood will be peaked (resp. spread out).

Influence of parameters. As we previously stated, two parameters are very important and hard to fix. Table 1 shows for example, how tracking errors can drastically increase depending on the choice of α value. We describe below their role.

The propagation parameter (σ) corresponds to the variance that defines how far from the current estimation the particles are propagated. If its value is too

high (resp. too small), then particles are propagated too far (resp. too close). Thus, the tracking will probably fail because only a small number of particles are well located in the state space.

The correction parameter (α) allows to adjust the spread of the likelihood. This is illustrated in Fig. 1, where the target is the person on the middle bottom of frames (surrounded by a red box). We show the likelihood maps (Bhattacharyya distance between color histograms, see Section 4) for different values of α. A white pixel (resp. black) correspond to a high (resp. low) likelihood value. As can be seen, when α value is small ($\alpha = 1$), large areas of the map have high values. Therefore, many particles get a high weight, even those far from the real object position. With such a small value for α the global estimation (weighted sum of particles) will probably be incorrect. For the case of $\alpha = 100$, the white areas are very small: only a small subset of particles will get high weight. If no particle were previously propagated onto this small high likelihood area, then the global estimation will also be incorrect.

Figure 1: Influence of the α parameter of the likelihood function. From left to right: the original image (the model is the histogram of the region surrounding the person on middle bottom), likelihoods maps with $\alpha = 1$ and $\alpha = 100$.

In the literature (Hassan et al., 2012; Maggio et al., 2007; Erdem et al., 2012; Pérez et al., 2002) these parameters are empirically fixed, depending on the tested sequences or the target application. Then, α is tuned to fix the spread of the likelihood so that its high value areas approximately cover the region surrounding the tracked object. Variances σ_w and σ_h are also fixed depending on the motion in the sequence: large values for high motion, and small ones for low motion. Moreover, these two parameters are defined for all the sequence: if the motion or appearance of the tracked object changes, as well as its surrounding context, the tracking will probably fail because these parameters are not well adapted anymore. In this paper, we propose an algorithm to adapt the propagation and correction parameters to each frame of the sequence without any additional cost in term of computation time. This is done by analyzing the dispersion of the particle set in order to determine if it is still adapted to the current context. If it is not adapted anymore, the parameters are changed to make this set suitable. Our approach is described in the next section.

3 PROPOSED APPROACH

In the particle filter framework, a resampling step can be necessary (step 5 of algorithm given in Section 2) in cases of degeneration of the particle set, *i.e.* when most of the particles have a low weight. It means the variance of the particle set is too high. Usually, the particle set is resampled when its associated efficient number becomes higher than a given threshold (Gordon et al., 1993). This number is computed from the normalized weights and monitors the amount of particles that significantly contribute to the posterior density. It is given by $N_t^{eff} = \frac{1}{\sum_{i=1}^{N}\left(w_t^{(i)}\right)^2}$. Note that, this number does not permit to distinguish a bad particle set (only low weights) from a very good particle set (only high weights). Indeed, for both cases, after the weight normalization, all particles get similar weights (around $1/N$), and the efficient number a similar value. This shows the difficulty to fix a general threshold on this efficient number to determine whether or not the particle set is correct and does not need to be resampled. However, it provides a good indication on the number of particles with high and low weights. As previously said, intuitively, we would like that most particles have high weights, but some also get low weights, to preserve the diversity of the particle set. For example, if a proportion of 0.5 of the particle set have high weights (around $2/N$ after normalization), the remainder with low weights (around 0 after normalization), then $N_t^{eff} \approx \frac{N}{2}$. Then, if a proportion of p particles have high weights, $N_t^{eff} \approx Np$. Here we call p the *survival rate*.

In this paper, we adopt an opposite reasoning than usually. Instead of adapting the particle set to the likelihood by resampling the former, we adapt the likelihood to the particle set. We search for the α value that provides a specific efficient number. Our idea is to determine which α value permits to reach this equality: $N_t^{eff} \approx \max_i(w_t^{(i)})$. This corresponds to a proportion $p \approx \frac{\max_i(w_t^{(i)})}{N}$ of particles with high weights. Fig. 2 shows two plots of the survival rate p (in red) and the maximum weight (in green) depending on the α value. As can be seen, the efficient number decreases when α increases because the number of particles with high weight usually decreases in cases of peaked likelihoods. On the contrary, the maximum weight value increases with α because high likelihood values are distributed in smaller areas. These curves have an intersection point, whose abscissa gives our optimal α value. The choice of $\max_i(w_t^{(i)})$ also permits to detect if the particle set was not well propagated, *i.e.* if σ does not suit the target's motion. If there is no inter-

section between p and $\max_i(w_t^{(i)})$ curves, it means, either the maximum weight has a very high value (close to an impoverishment problem), or a low one (close to the degeneracy issue). In this case, we increase σ value and re propagate particle set, and then search again for the optimal α. Our algorithm is summarized in Algorithm 1. Here $BD(.,.)$ is the Bhattacharyya distance between the model and particle descriptors.

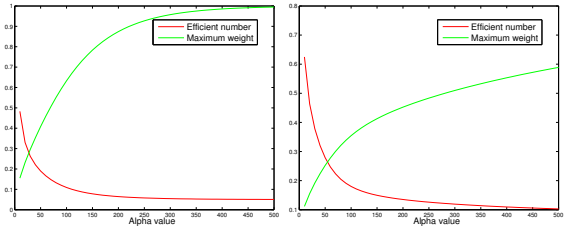

Figure 2: Plots of survival rate and maximum weight values depending on α value for a particle set ($N = 20$), between frames 1 and 2 of sequences `Crossing` and `Liquor`.

Input: Image I_t, particle set $\{\mathbf{x}_{t-1}^{(i)}, w_{t-1}^{(i)}\}_{i=1}^{N}$
Particle set $\{\mathbf{x}_t^{(i)}, w_t^{(i)}\}_{i=1}^{N}$, α_{init}, σ_{init}
α_{max}, α_{update}, σ_{max}
Output: Optimal α value α_{opt}

$diff \leftarrow 1$, $\alpha \leftarrow \alpha_{init}$, $\sigma \leftarrow \sigma_{init}$
$\mathbf{x}_t^{(i)} \sim \mathcal{G}(\mathbf{x}_{t-1}^{(i)}, \sigma)$
$D_t^{(i)} \leftarrow BD^2(.,.)$, $i = 1,\ldots,N$
while $diff > 0$ *and* $\alpha \leq \alpha_{max}$ *and* $\sigma \leq \sigma_{max}$ **do**
  $w_t^{(i)} \leftarrow e^{-\alpha D_t^{(i)}}$, $i = 1,\ldots,N$
  $p = \frac{1}{N\sum_{i=1}^{N}\left(w_t^{(i)}\right)^2}$
  $diff = p - \max_i(w_t^{(i)})$
  if $\alpha = \alpha_{max}$ **then**
    $\sigma \leftarrow 2\sigma$
    $\mathbf{x}_t^{(i)} \sim \mathcal{G}(\mathbf{x}_{t-1}^{(i)}, \sigma)$
    $D_t^{(i)} \leftarrow BD(.,.)$, $i = 1,\ldots,N$
    $\alpha \leftarrow \alpha_{init}$
  $\alpha \leftarrow \alpha + \alpha_{update}$
$\alpha_{opt} \leftarrow \alpha - \alpha_{update}$

Algorithm 1: Algorithm for the selection of optimal α for correction and σ adaptation for propagation.

4 EXPERIMENTAL RESULTS

In this section, we show the efficiency of our proposed algorithm to increase tracking accuracy. We first give our experimental setup in Section 4.1. Quantitative results are then given in Section 4.2. Finally, 4.3 propose a study of the interest of our approach in complex tracking situations and its convergence, stability and computations times.

4.1 Experimental setup

In this paper, we use the **same** experimental setup for all tests provided. This setup is given below.

The state vector contains the parameters describing the object to track, and is defined by $\mathbf{x}_t = \{x_t, y_t\}$, with $\{x_t, y_t\}$ the central position in the current frame of the tracked object. A particle $\mathbf{x}_t^{(i)} = \{x_t^{(i)}, y_t^{(i)}\}$, $i = 1,\ldots,N$, is then a possible spatial position of this object.

The observation model is computed into a fixed size region surrounding the state position. This model is manually initialized in the first frame of the sequence (from the ground truth associated with the tested sequences - see Section 4.2). We have tested two different descriptors for surrounding regions. The first one, called $\mathcal{M}_{col}$, is the color histogram. It is build by concatenating the 8-bin histograms of the three RGB channels to get a 24-bin histogram. The second one, $\mathcal{M}_{col+hog}$ contains both color and shape information. The color information is the same as in $\mathcal{M}_{col}$. The shape information is the concatenation of two 8-bin histograms of gradient (HOG) (Dalal and Triggs, 2005) computed in the upper and bottom parts of the region. Then, we get two histograms for $\mathcal{M}_{col+hog}$: a 24-bin color histogram plus a 16-bin HOG. These descriptors will be used for the observation model and each particle's region.

Particles are propagated using a Gaussian random walk whose variance is fixed depending on the size of the region surrounding the tracked object. Knowing h and w respectively the height and width of this region (fixed and given by the ground truth), we use $\sigma_w = w/2$ and $\sigma_h = h/2$ as propagation parameters (note that in our algorithm, σ is adapted over time if necessary).

Particles' weights are computed in two different ways, depending on the descriptor. For $\mathcal{M}_{col}$ descriptor, we compute $w_t^{(i)} \propto e^{-\alpha \mathbf{d}_{col}^2}$, where $\mathbf{d}_{col}$ is the Bhattacharyya distance between the color histogram of the model and the one of the target. For descriptor $\mathcal{M}_{col+hog}$, we compute $w_t^{(i)} \propto e^{-\alpha(\mathbf{d}_{col}\times\mathbf{d}_{hog})^2}$, where $\mathbf{d}_{hog}$ is the Bhattacharyya distance between the model's HOG and particle's one.

The current estimation of the tracked objects' position is the weighted sum of particles positions.

A multinomial resampling (Gordon et al., 1993) is done at the end of each time step.

For all tests, we fixed $N = 20$. Algorithm 1 also requires to fix some parameters concerning the initialization, update and maximum value of α. We always

use the same parameters: $\alpha_{init} = 10$, $\alpha_{update} = 10$, $\alpha_{max} = 500$, $\sigma_{max} = 10$. $\mathcal{M}_{ref}$ is the descriptor of the model *i.e.* object to track and $\mathcal{M}_t^{(i)}$ is the one of the i^{th} particle in the frame t. *diff* is the criterion to determinate the intersection between the two curves.

4.2 Quantitative results

In order to evaluate our approach, we use the Tracker Benchmark from (Wu et al., 2013). It contains 50 challenging sequences with various visual tracking difficulties (illumination changes, deformation, rotation, scale changes of the target and the context, *etc.*). All sequences are provided with their ground truth, that permits to perform a quantitative evaluation.

We compare tracking errors obtained with our approach, with a classical approach that uses fixed values used for α (20, 50, 100 and 200). We also compare our method with the one in (Brasnett and Mihaylova, 2007), *i.e.* $\alpha = \frac{1}{\sqrt{2d_{min}}}$. We tested the two descriptors $\mathcal{M}_{col}$ and $\mathcal{M}_{col+hog}$. Table 1 gives tracking errors obtained by all methods on some sequences of the benchmark (all sequences have been tested), corresponding to a mean over 10 runs. This table shows our approach gives most of times lower tracking errors (in bold) for both models. Overall the sequences, our tracking errors, compared to other approaches, are from 13% to 20% lower with descriptor $\mathcal{M}_{col}$ and from 24% to 33% lower with descriptor $\mathcal{M}_{col+hog}$. We can also highlight that for fixed α, lower tracking errors are never get with the same α value (see underlined scores), showing there is no universal α value. The approach suggested in (Brasnett and Mihaylova, 2007) sometimes gives lower tracking errors than with all fixed α, but often a specific fixed α value achieves better results. However, it does not achieve as lower errors as our approach does, except for two cases (`Trellis` and `Skiing` sequences).

Furthermore, we note some cases where our approach does not give the lowest tracking errors and they can be classified into two main classes. First, cases where our tracking errors are not the best but very close to the lower tracking error (see for example sequences `Mhyang` or `Freeman1`). Secondly, cases of divergence of the particle filter: for all approaches including ours, tracking errors are very high. Indeed, the benchmark contains very complicated sequences, such like `Ironman` for example (moving camera, high deformations, strong motions, occlusions, *etc.*).

Moreover, we have compared the tracking errors obtained with our method and errors obtained for fixed α values ranging from 10 to 500 (step of 10). Results are given in Fig. 3 for two sequences (`Football1`, and `Matrix`). Each plot are tracking error (mean over 10 runs) obtained with different fixed value for α (blue curves). One can see errors obtained with our approach (red lines) are always lower than the lower error given using a specific and fixed α value. These plots show how much the errors with fixed α are not stable. This, once again, demonstrates the impact of the α value and the difficulty of choosing a good one adapted for a whole sequence and even more generally to a set of sequences.

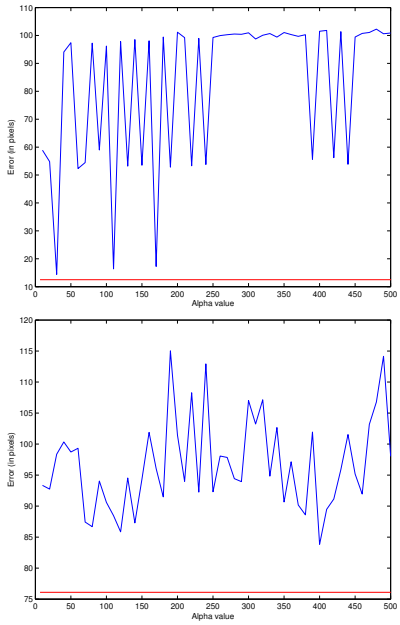

Figure 3: Variation of tracking errors ($N = 20$), depending on the value of α (fixed for all the frame). In red: tracking error obtained with our approach. From top to bottom, sequences `Football1` and `Matrix`.

All these results point out that the online modification over the time of the α value allows our algorithm to decrease tracking errors. In the next section, we will show how our self-adaptive likelihood function performs in specific tracking context.

4.3 Complex tracking conditions

Proximity between similar color objects. Figure 4 gives the best α values selected at each frame by our algorithm for the `Crossing` sequence, in which we attempt to track the person on the bottom (using $\mathcal{M}_{col}$). As can be seen on this figure, during frame interval $[20, 50]$, α gets high values, while after and before this interval time it gets low values. Indeed, in this sequence, during this interval of time, a car gets very close to the tracked person (see on the bottom of this figure, frames 20, 40 and 50). As its color is similar, a large region of the likelihood around the tracked peo-

Table 1: Tracking errors obtained on some sequences among the 50 of Tracker Benchmark proposed in (Wu et al., 2013) for different approaches. Lower tracking errors are in bold and lower given by a fixed α are underlined.

	$\mathcal{M}_{col}$: Color histogram						$\mathcal{M}_{col+hog}$: HOG + Color histogram					
Sequence	Our α adapt.	$\alpha = \frac{1}{\sqrt{2}d_{min}}$	$\alpha =$ 20	$\alpha =$ 50	$\alpha =$ 100	$\alpha =$ 200	Our α adapt.	$\alpha = \frac{1}{\sqrt{2}d_{min}}$	$\alpha =$ 20	$\alpha =$ 50	$\alpha =$ 100	$\alpha =$ 200
Sylvester	**18.1**	23.6	19.0	19.1	**18.1**	18.5	**8.2**	28.8	10.4	8.9	9.1	**8.2**
Trellis	71.6	**64.1**	71.9	78.2	78.0	80.6	**66.0**	72.3	119.5	70.4	78.5	78.9
Fish	**34.5**	52.3	37.5	34.8	**34.5**	35.3	**23.7**	41.5	30.8	27.9	24.6	25.1
Mhyang	16.5	32.1	16.9	**16.3**	16.4	16.7	**14.6**	23,7	15.4	14.8	29.3	29.0
Matrix	**76.1**	79,1	105.7	87.6	95.7	91.9	**64.4**	71.2	82.2	90.2	100.2	78.3
Ironman	**118.9**	148.4	135.3	147.4	158.6	159.0	114.3	154.8	143.3	178.2	**113.0**	200.6
Deer	**46.0**	92.9	75.7	86.2	50.3	57.8	**59.1**	81.6	77.8	112.6	210.4	59.7
Skating1	**68.0**	91.3	103.7	85.8	91.1	89.5	**44.2**	66.2	81.9	51.3	57.2	83.5
Boy	**32.8**	72.3	39.5	48.2	63.4	60.6	**14.1**	53.7	44.0	33.3	33.5	34.8
Dudek	**71.8**	161.4	110.3	80.5	84.3	85.4	**44.2**	101.2	91.9	161.4	48.4	58.0
Crossing	**7.7**	86.7	23.3	8.8	8.6	8.5	**5.1**	29.3	7.6	5.8	5.4	5.6
Couple	**15.6**	28.8	35.3	17.9	17.3	17.2	**14.2**	19.5	20.1	62.1	16.8	18.2
Football1	**12.5**	53.9	25.8	17.7	14.9	15.4	**11.7**	28.2	20.7	18.2	13.6	13.1
Freeman1	43.5	99	46.8	62.1	**42.9**	56.3	**15.0**	18.4	16.4	15.6	15.3	19.2
Jumping	**42.9**	53.5	50.8	46.4	44.4	50.9	**54.7**	68.5	100.7	60.2	57.7	56.0
CarScale	**47.4**	47.5	59.6	60.9	48.2	54.1	**33.6**	51.2	62.1	59.1	60.4	45.2
Skiing	238.9	**178.0**	237.3	263.0	266.9	276.2	211.2	**166.1**	196.1	227.0	264.9	225.8
Dog1	**17.1**	24.8	21.4	18.9	17.5	17.6	**47.4**	48.0	50.0	50.0	50.4	49.5
MoutainBike	**22.2**	84.6	28.8	40.0	30.9	35.6	**11.3**	96.0	20.3	18.0	11.9	17.5
Lemming	**43.6**	127.0	122.3	81.5	91.9	78.2	**101.8**	125.2	209.6	138.2	119.3	244.2
Liquor	**46.4**	53.1	69.1	59.2	54.3	57.1	**38.2**	45.9	67.5	52.3	43.3	42.4
Faceocc2	**48.1**	57.9	51.4	63.0	54.6	55.1	**35.8**	49.2	41.7	59.4	58.9	41.0
Basketball	**83.1**	85.6	86.5	87.1	85.8	83.7	**40.4**	62.2	184.5	99.3	82.4	54.6
Football	**118.5**	122.6	178.0	170.6	203.8	210.5	**109.6**	118.2	185.9	219.5	227.1	234.0
Mean (over 50 seq.)	**62.3**	78.7	78.7	75.2	74.5	71.8	**49.6**	64.4	74.4	69.6	65.2	66.4

ple gets high values. Usually, the tracking with a fixed value for α would be disturbed, because a lot of particles will get high weights, even those far from the tracked person, or on the other object with a similar color. To better track the person, it is then necessary for the likelihood to be more peaked on the tracked object, that is the reason why α should get high values during this time. We see on this example that our approach is able to adapt to the context (*i.e.* two objects with similar colors close to each other).

Middle plot of this figure corresponds to tracking error curves obtained with our algorithm (red curve), and fixed values over all the sequence, $\alpha = 20$ (blue curve), $\alpha = 50$ (green curve) and $\alpha = 100$ (black curve). Thanks to our automatic selection of α value, our tracking errors decrease during interval time $[20, 50]$, contrary with those obtained with fixed α. Globally, on `Crossing` sequence, our approach provides an average error of 7.7%, while errors with $\alpha = 20$, $\alpha = 50$ and $\alpha = 100$ are respectively 23.3%, 8.8% and 8.6%.

Illumination changes. At the end of the `Crossing` sequence (from approximately frame 80), there is an illumination change. However, our approach is less disturbed: its errors are the lowest, while they can drastically increase with a bad fixed value for α (see for $\alpha = 20$ for example). `Fish` sequence shows a toy slowly moving under a light (see some frames on the bottom of Fig. 5). The illumination conditions are hardly changing, and the color model, fixed over time, is not adapted to efficiently track the object during the sequence. Fig. 5 presents the evolution of optimal α values over time, during frames $[200 - 300]$. As can be seen, its value often change with time, due to the changes in illumination. Tracking errors are shown in the middle plot of this figure: our method (red curve) is more stable with time than the others. This illustrates the interest of our approach to better track with time in such conditions.

These tests have proven that our automatic adaptation of the α value allows to improve tracking performances (i) in case of proximity between similar color objects and (ii) in case of illumination changes. Indeed, in such cases, most of times the modes of the likelihood can become very spread out or peaked: increasing or decreasing α value permits to adapt our correction step to the current context.

Convergence study. All our previous tests were made with $N = 20$ particles, this is important to check if our approach does not get the best results only with

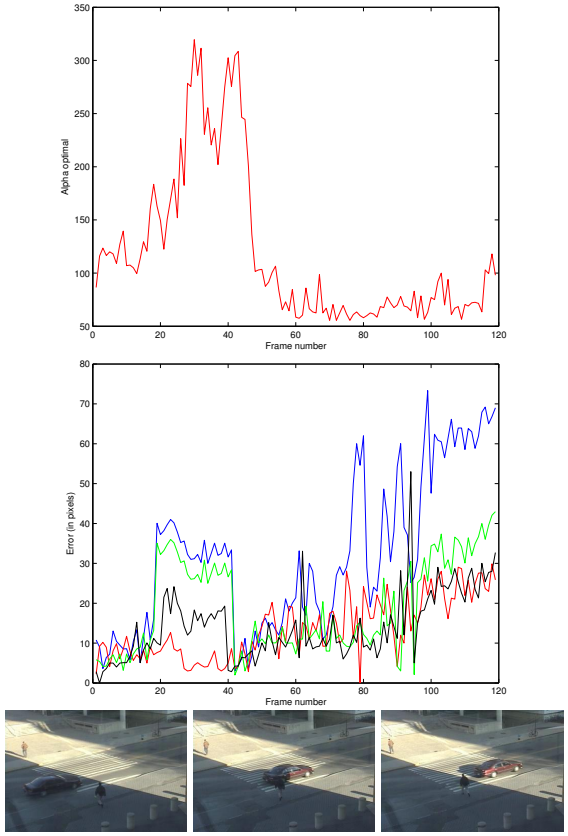

Figure 4: `Crossing` sequence. Top: optimal α selected by our approach along the sequence (120 frames, mean over 30 runs). Middle: tracking errors obtained with our approach (red) and with fixed $\alpha = 20$ (blue), $\alpha = 50$ (green) and $\alpha = 100$ (black). Bottom: frames 20, 40 and 50, *i.e.* before, during and after the car and the walking tracked person trajectories cross.

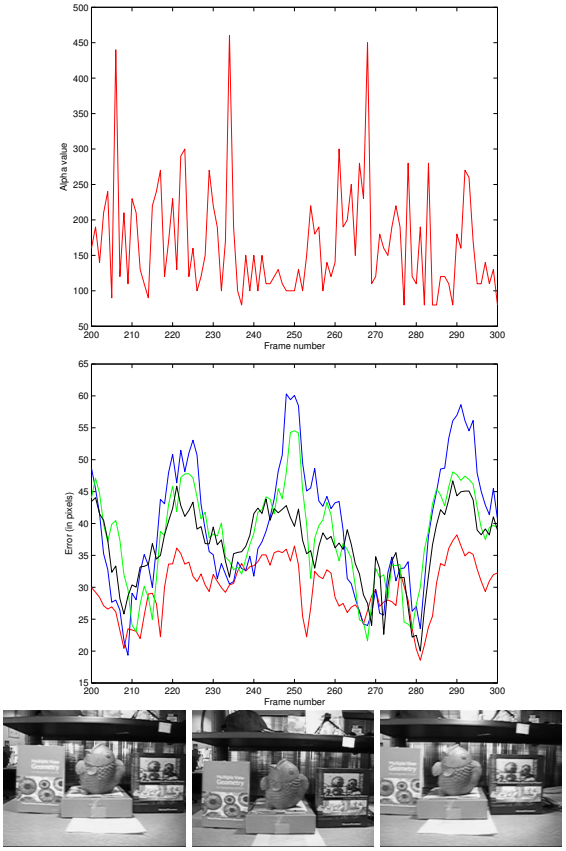

Figure 5: `Fish` sequence (frames 200 to 300). Top: optimal α selected by our approach along the sequence (100 frames out of 476, mean over 30 runs). Middle: tracking errors obtained with our approach (red) and with fixed $\alpha = 20$ (blue), $\alpha = 50$ (green) and $\alpha = 100$ (black). Bottom: frames 205, 240 and 290, *i.e.* when object is lighted or not.

a small number of particles. We then give in Fig. 6 convergence curves, *i.e.* errors depending on the number of particles for two video sequences (`Walking` and `Couple`). In these plots, we compare results obtained with fixed α (20, 50 and 100) values and our adaptive α. We can see our approach always converges faster and better. This proves we could use less particles to achieve lower tracking errors than with fixed α.

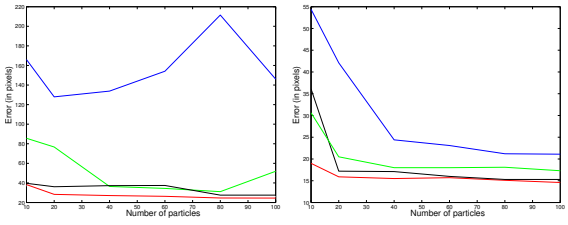

Figure 6: Convergence curves according to the number particles for `Walking` and `Couple` sequences. In red our approach, in blue $\alpha = 20$, in green $\alpha = 50$, in black $\alpha = 100$.

Stability. We previously explained that our approach which adapts the correction step permits a better and faster convergence of the tracker compared to approaches with a fixed α value. We reported in Table 2 some comparisons of variances obtained over 10 runs with descriptor $\mathcal{M}_{col}$. As can be seen, our algorithm also reduces the variance over the runs, that proves its stability. Indeed, the variance over the runs is highly dependent on the random generation of samples (propagation): if the set of particles is badly propagated, a particle filter with a fixed α will probably fail in correctly tracking. On the contrary, our approach permits α to adapt the likelihood to the propagated particle set, that makes it more stable. Note that when using $\alpha = \frac{1}{\sqrt{2}d_{min}}$ heuristic, we get similar or lower variances than with fixed α, but not as small as our variances.

Table 2: Stability (variance of tracking errors over 10 runs) obtained with fixed and adaptive α on several video sequences. The lowest in bold.

Sequence	Our α adapt.	$\alpha = \frac{1}{\sqrt{2}d_{min}}$	$\alpha =$ 20	$\alpha =$ 50	$\alpha =$ 100	$\alpha =$ 200
CarDark	**0.4**	12.8	23.5	20.0	22.7	18.9
Fish	**0.2**	0.6	0.7	0.3	0.4	0.4
Matrix	**7.9**	14.6	15.3	12.7	18.8	9.5
Couple	**1.2**	3.1	2.5	1.5	2.4	2.0
CarScale	**4.3**	7.1	6.3	5.6	5.1	7.2
Football	**2.2**	29.8	93.5	41.1	34.8	43.9
Dog1	**2.3**	5.1	3.9	4.2	3.9	10.2

About computation times. Our tests show that our approach does not add an additional computational cost. Indeed, computation times are similar whether we use a fixed α or adapt it over time. In particle filter, the most time consuming step is the correction one, more precisely the computation of distances between particles and model. This step is done once, for fixed or adaptive α. Our algorithm is just a loop over all possible values for α (form 10 to 500, with a step of 10), that corresponds to a maximum of 50 values. This is why our optimal α does not add any cost to the global particle filter algorithm. Note that sometimes our computation times are lower: because our likelihood density is better adapted to the particle set, the resampling can require less time. We get an average of $+0.42\%$ of the computation times for all 50 tested sequences.

5 CONCLUSION

In this paper, we have presented an approach that automatically adapts over time fundamental parameters of the likelihood function of the particle filter. More precisely, using a single and simple criterion, it can set the correction parameter as well as the propagation parameter. Moreover, it does not require any additional cost in term of computation times.

Our tests have proven on several and challenging video sequences the high impact of these two parameters on the tracking performances. However, they are often neglected and set up with fixed and arbitrary values that can, as we have shown, increase the tracking errors. Our experiments also show, that our method which adapts these parameters depending on the context greatly improves the robustness of the particle filter. Moreover, it converges better, faster and is more stable. Particularly, our algorithm still succeeds in complex tracking situations like illumination changes or proximity between similar color objects.

Further works will concern the validation of our approach by using different kinds of descriptors (such as wavelets), similarity measures (such as Chamfer) in order to prove the generalization of our technique. We also attempt to show our approach can also improve multi-cue or multi-modal tracking accuracies. Finally, we are working on the derivation of a mathematical demonstration of the validity of our criterion used to determine the optimal correction value.

REFERENCES

Bhattacharyya, A. (1943). On a measure of divergence between two statistical populations defined by their probability distributions. *Bulletin of Cal. Math. Soc.*, 35(1):99–109.

Brasnett, P. and Mihaylova, L. (2007). Sequential monte carlo tracking by fusing multiple cues in video sequences. *Image and Vision Computing*, 25(8):1217–1227.

Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *CVPR*, pages 886–893.

Erdem, E., Dubuisson, S., and Bloch, I. (2012). Visual tracking by fusing multiple cues with context-sensitive reliabilities. *Pattern Recognition*, 45(5):1948–1959.

Fontmarty, M., Lerasle, F., and Danes, P. (2009). Likelihood tuning for particle filter in visual tracking. In *ICIP*, pages 4101–4104.

Gordon, N., Salmond, D., and Smith, A. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. of Radar and Signal Processing*, 140(2):107–113.

Hassan, W., Bangalore, N., Birch, P., Young, R., and Chatwin, C. (2012). An adaptive sample count particle filter. *Computer Vision and Image Understanding*, 116(12):1208–1222.

Lichtenauer, J., Reinders, M., and Hendriks, E. (2004). Influence of the observation likelihood function on object tracking performance in particle filtering. In *FG*, pages 767–772.

Maggio, E., Smerladi, F., and Cavallaro, A. (2007). Adaptive Multifeature Tracking in a Particle Filtering Framework. *IEEE Transactions on Circuits and Systems for Video Technology*, 17(10):1348–1359.

Ng, K. K. and Delp, E. J. (2009). New models for real-time tracking using particle filtering. In *VCIP*, volume 7257.

Pérez, P., Hue, C., Vermaak, J., and Gangnet, M. (2002). Color-Based Probabilistic Tracking. In *ECCV*, pages 661–675.

Wu, Y., Lim, J., and Yang, M.-H. (2013). Online object tracking: A benchmark. In *CVPR*, pages 2411–2418.