Fast Color-Texture Discrimination: Application to Car Tracking

John Klein, Christele Lecomte and Pierre Miche
LITIS lab, University of Rouen,
76800 St Etienne du Rouvray, France
john.klein@etu.univ-rouen.fr

Abstract— **Visual tracking methods have been intensively contributed in the past decade. Promising results have been brought, leading to partial solution of the problem. However it is still utmost difficult to maintain track of an object for a long time, because some events can strongly disrupt the tracking procedures. Such events are occlusions, clutters, illumination changes, particular movements or pose changes. To overcome these challenging events and produce more reliable tracking algorithms, image data must be exploited through several aspects, that is to say through several cues : texture, color, shape or movement. But before being able to use these sources, one must make sure that each of these sources is reliable and non-redundant. In this article, we reckon that texture and color must be jointly processed, and we propose a new color-texture feature called weighted cooccurrence matrices. Using this feature within a particle filter, successful car tracking examples are proposed.**

I. INTRODUCTION

Object tracking is one of the most challenging problems of computer vision. In the past decade great improvements were performed, and some tracking algorithms already gave remarkable results. Nowadays, the research efforts focus on enhancing the existing approaches, so as to one day be able to build a global fully robust solution to the problem. Indeed many events occurring in natural scenes can disrupt algorithms : occlusions, clutters, illumination changes, particular movements or pose changes. Many directions can be investigated to overcome the failures coming from such events, yet experts from computer vision, as well as from pattern recognition, all stress the importance of reliable features in entry to any higher order procedure.

In this article we focused on color and spatial information, which can jointly be processed by extracting a color-texture feature. To produce such a feature, some authors firstly thought about applying texture extraction techniques to each color plan. Jung [9] computes wavelets on RGB plans, and so does Palm [11] with cooccurrence matrices. It is also possible to use texture extraction by substituting gray-level with colors, if you consider color as a continuous quantity. These approaches must be carefully designed, because they induce closeness between colors, which may not be relevant. Chang [3] calculates color cooccurrences, by counting occurrences of neighbor colors, instead of neighbor gray levels, and thereby induces no sense of closeness. Other methods do not belong to the two previously cited schemes, like color coherence vectors, spatial chromatic histograms [4], color density [5], as well as the works of Paschos [12].

Color cooccurrences were retained as a first processing, because spatial and color information are equally rendered by this feature. Yet our purpose in this article is to add property to such feature, so that it matches tracking purposes. Occlusions or clutters cannot be handled by a feature extraction technique because they depend on higher order concepts such as object, scene or context, but not on pixels values. Pose changes do not depend on feature extraction methods efficiency, but on learning accuracy. At least, color and texture should be learnt on several faces of the tracked object to overcome this difficulty, but this is not our aim in this article. Existing features, inluding cooccurrences, already can partially deal with movement invariance.

Our contribution in this article allows our new feature to cope with soft illumination changes, that are inevitable in any natural scene. The principle relies on the addition of weights for each color, using kernels centered on colors of the target object. This new feature is by essence more adapted to tracking issues, because it will be able to cope with soft illumination changes by its own. To judge the efficiency of our approach, two tracking aspects will be looked over : precision and robustness. To carry out the tests, the feature will be used inside a particle filter. Particle filters have proven to be reliable tracking methods, and make no restrictive hypotheses about the object model, so they are adapted to feature evaluation.

This article is organized as follows, the first part will present our contribution. Calculation of weighted color cooccurrences will be detailed and justified. As a second part, a particle filter using our color-texture feature for visual car tracking will be presented. In the third section some experiments will be presented and discussed. Finally results and procedure are summarized in a conclusion.

II. WEIGHTED COOCCURRENCE MATRICES

A. Cooccurrences

Cooccurrence matrices constitute one of the most popular texture characterization techniques. They were introduced by Haralick [8] for gray-level texture. The formula used for their computation is the following :

$$M_{\vec{d}}(i,j) = \frac{\#\left\{p \in D / I(p) = i, I\left(p+\vec{d}\right) = j\right\}}{\#\{D\}} \quad (1)$$

I is an image defined on D, p a pixel of this image, $\vec{d}$ a translation vector, (i,j) a couple of gray levels and $\#$ is the cardinal of a set. It simply consists in storing statistics about grey levels topology. Note that in addition the computation of the matrices is fast. The vectors $\vec{d}$ are generally chosen so as to draw statistics in the 8-neighborhood of the pixel. This leads to four directions imposed by the discrete space D, ie $\vec{d} = \begin{pmatrix} cos\theta \\ sin\theta \end{pmatrix}$ with $\theta \in \left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\right\}$. Examining a texture at different scales may impose to adapt the norm of $\vec{d}$.

Adaptation to color is natural, the couple of gray levels can be replaced by a couple of colors. Thus matrices characterize colors, because if a color is absent from the processed texture, its line and column within the matrix will be empty. Spatial information is depicted by the number of occurrences of a given couple of color.

B. *Adding weights*

As part of car-tracking application, cooccurrence matrices from an image subwindow containing the car will constitute an a priori model. To perform the track in the driving sequence, we will afterwards need to judge the resemblance of any subwindow from the current image with our model. Such judgment can be obtained by comparing matrices from the local subwindow to those of the model, therefore a metric for the matrices is required. Cooccurrence matrices can be considered as histograms, because the storing order of the statistics bears no importance. Many existing distances, or pseudo-distances, can be used for histograms : classical L_1, L_2 or L_p Minkowski distances, Kullback-Leibler divergence, χ^2 distance, Hellinger distance or Bhattacharyya distance. The latter one was retained for our study, because it is well-known for its noise robustness [5]. Distance d is computed from two normalized histograms h and h', whose bins are indexed by i :

$$d\left(h, h'\right) = 1 - \sum_{i=1}^{M} \sqrt{h_i h'_i} \tag{2}$$

The two histograms must have the same number of bins M. Such distance can be applied to the cooccurrence matrices, after concatenation. Using this distance, it is clear that only identical matrices will produce a null distance. However, imagine that the matrices are computed upon two different images of the same car, then only a slight change of illumination will produce very different matrices, therefore leading to a distance closer to 1. In other words, cooccurrences precision is a drawback for tracking purpose, because one object can produce some matrices at some times of the sequence, which cannot be matched to the reference ones.

Noticing this problem, we came up with the idea of changing the occurrences increment. Indeed the goal to reach is that colors of the same kind fall in the same bins, so as to cope with slight illumination changes, that are unavoidable in natural scenes. A naive solution would be to reduce the number of bins M, but if so, the grid of the matrices is not representative of the car color-texture. The proposed solution is to give different voting weights to each color c_i, depending on whether this color is close to a representative color of the car, denoted as $\tilde{c_j}$. The sets of representative colors can be limited to a very few colors, considering cars, the color of the body is the most important, but red can be added for rear lights and black for tires. We propose to compute the weights w_i using kernels centered on the representative colors :

$$w_i = \sum_{j=1}^{N_K} K_j \left(\|\tilde{c}_j - c_i\|\right) \tag{3}$$

N_K is the number of kernels equal to the number of representative colors, and $K_j()$ is a kernel. Kernels are adequate for our problem, because colors very close to a chosen representative color will be almost as important as the chosen one, colors a little far from the chosen ones will influence the increment and finally colors too far from the model will all be silent. In our study we have considered the use of three different kernels displayed over one dimension on figure 1. These kernels mix more or less data. The most

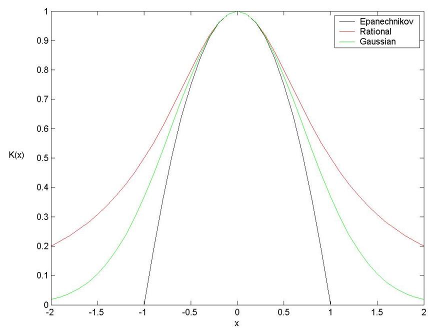

Fig. 1. 1D representation of several kernels

mixing one is the rational kernel, the least mixing one is the Epanechnikov kernel [5], and the compromise is the gaussian kernel [14]. It is interesting to mix data if we want several colors to have similar weights, even if this means adding some noise.

The choice of the norm, or distance cf. equation 3, used for comparing colors before running it through a kernel can be discussed. We chose simple euclidian distance between RGB coordinates. There may exist better distances, which can take into account perceptual closeness of colors, but that is more or less equivalent to changing the color space. The choice of the color space relative to illumination invariance is beyond the scope of this paper, as a consequence we restricted ourselves to the usage of euclidian distance in the RGB space. The efficiency of the weighted color cooccurrence matrices (WCCM) for each different kernel will be examined in section IV.

III. APPLICATION TO CAR TRACKING

To be able to judge the weighted cooccurrence matrices feature, it must be used inside a tracking procedure. Particle filters were retained as the tracking procedure to use for the tests, because they are famous for their robustness, and also because they induce no constraints on the object modeling. In this section a short review of particle filters techniques is given, as well as the implementation of it for car tracking.

A. Sequential Monte Carlo methods

The goal of particle filters, also known as sequential Monte Carlo methods, is to estimate a random vector X_t, referred as the state vector. This estimation becomes possible knowing a second random vector Y_t, referred as the observation vector. As in many inference problems, the estimation of X_t is obtained using the estimation of the filtering density $p\left(X_t|Y_{1:t}\right)$, where $Y_{1:t}$ corresponds to all available observations at time t.
To perform such an estimation, some information is required. First of all, the equations binding the state to its previous realization and to the observation are required.

$$X_t = F\left(X_{t-1}, W_t\right) \tag{4}$$
$$Y_t = H\left(X_t, V_t\right) \tag{5}$$

F and H are two non-linear functions and W_t and V_t are two noises. As for a Kalman filter F and H should be linear and W_t and V_t should be gaussian, that is why particle filter is more powerful than Kalman filter, even if Kalman filter is an exact method.
To perform the estimation the particle filter theory uses many results of the probabilistic theory. The first one is the Monte Carlo approximation, derived from the law of great numbers :

$$\hat{p}\left(X_t|Y_{1:t}\right) = \sum_{i=1}^{N} \delta_{X_t^{(i)}}\left(X_t\right) \tag{6}$$

$\left(X_t^{(i)}\right)_{i:1..N}$ are N samples, or particles, drawn from a law $p\left(.|Y_{1:t}\right)$, $\delta_{X_t^{(i)}}$ is the Dirac distribution centered on the particle $X_t^{(i)}$.
The second result to be exploited is importance sampling. Indeed it is unreasonable to think that one can sample from p at any time. To overcome this difficulty, importance sampling proposes to sample from another law q, whose support must contain the support of p. Then, the samples will have to be weighted by $w_t^{(i)} = \frac{p(X_t|Y_{1:t})}{q(X_t|Y_{1:t})}$. Finally the estimation becomes :

$$\hat{p}\left(X_t|Y_{1:t}\right) = \sum_{i=1}^{N} w_t^{(i)} \delta_{X_t^{(i)}}\left(X_t\right) \tag{7}$$

In addition, under markovian hypotheses, it can be shown that a recursive estimation of weights is possible using the formula :

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{p\left(Y_t|X_t^{(i)}\right) p\left(X_t^{(i)}|X_{t-1}^{(i)}\right)}{q\left(X_t^{(i)}|X_{t-1}^{(i)}, Y_t\right)} \tag{8}$$

The tracking procedure is now complete, but it is generally added one more step : the resampling step, which allows particle to be renewed, thereby avoiding the degeneracy observed in the simple particle filter procedure. This step was added by Gordon [7], and made the development of particle filter popular. For more details about particle filters, and resampling techniques, please refer to these works [2], [6].

B. Particle filter for car tracking

The first step is to design a state vector matching our tracking purpose. Among the literature two representations are commonly used : the bounding box and the contour. In the case of the bounding box, the location of target center along with the box dimensions compose the state. In the case of contour, a spline is often used, and the control points of the spline constitute the state. In our approach, we will choose the bounding box state representation, which is easier to manipulate :

$$X_t = \begin{pmatrix} i_t \\ j_t \\ height_t \\ width_t \end{pmatrix} \tag{9}$$

The center of the target is given by (i_t, j_t), and its dimensions are $(height_t, width_t)$. A particle is consequently a template for image subwindow. Different sizes and positions of subwindows will be generated by the filter.
The state evolution equation is often modeled as an AR-2 process, depending on both previous state and its velocity, in this article we will restrict to a simple random walk :

$$X_t = X_{t-1} + \begin{pmatrix} \mathcal{N}\left(0, \sigma_1\right) \\ \mathcal{N}\left(0, \sigma_2\right) \\ \mathcal{N}\left(0, \sigma_3\right) \\ \mathcal{N}\left(0, \sigma_4\right) \end{pmatrix} \tag{10}$$

In addition this random walk will be chosen as the proposal density $q = p\left(X_t|X_{t-1}\right)$, which is called the prior density. The prior density simplifies the calculation of the weights $w_t^{(i)} \propto w_{t-1}^{(i)} p\left(Y_t|X_t^{(i)}\right)$. In these conditions, the random walk also has the advantage of making no hypothesis about the object movement, therefore the sampling of the particles is not restricted to some direction. It would be of course in terms of tracking efficiency more interesting to sample particles in the image, where the object is more likely to lie. This is what is proposed by famous particle filters implementations [15] [10]. Moreover the best choice for the proposal density is $p\left(X_t^{(i)}|X_{t-1}^{(i)}, Y_t\right)$, see [6] for a proof and [1] for an example of possible usage of the optimal proposal density. Yet our purpose in this article is not to design a better particle filter, but to design a better texture characterization, so we want the algorithm efficiency to mostly depend on that.
The nature of the observations Y_t depends on the chosen features, but they all take their origins within the image that is observed at time t. In our case this feature is weighted cooccurrence matrices. A model for the observation equation is needed, which is equivalent to proposing a likelihood $p\left(Y_t|X_t\right)$. This likelihood can be understood as the probability to observe the texture of the tracked car in the state-corresponding subwindow, that is observed. To match this idea, we propose the following expression of the likelihood :

$$p\left(Y_t|X_t\right) \propto \exp\left[-\lambda d_{min}\left(Y_t, Y_{Ref}\right)^2\right] \tag{11}$$

Y_{Ref} is the set of reference matrices learnt on a subwindow containing the car. Y_t is the set of matrices computed on the local subwindow. d_{min} is the minimal Bhattacharyya distance between matrices of the two sets. Parameter λ helps to strengthen the difference between subwindows containing background, like road for instance, and subwindows containing the car. If subwindows containing road must return a likelihood up to $0,01$ for instance, then the value of λ is fixed.

C. Implementing weighted cooccurrences matrices

In this sub-section, a WCCM computation method in reasonable time is proposed. A matrix computation time is linear with respect to the analyzed image or subwindow size. For each pixel N_K weights must be calculated, whose cost depends on the chosen kernel. These costs can be reduced to a *read operation* time by generating a lookup table. This table will contain precalculated quantified values of the chosen kernel. Thus the overall computation time will decrease, even if it will require more memory space.
Let N_q be the number of quantified colors, the matrix size is then qxq. To maintain heterogeneity of colors one should choose q between 256 and 128. Only a few colors are selected to represent the object texture, so many cells of the matrix will be empty. Memory space and computation time can be saved by discarding the matrix zeros.
Let c_{ref1} be the closest reference color to an observed color c_i. Let c_{ref2} be the closest reference color to a second observed color $c_{i'}$ distant from c_i by $\vec{d}$. Since the cell corresponding to $(c_i, c_{i'})$ is incremented by $W_i W_{i'}$, it turns out to be simplier to increment (c_{ref1}, c_{ref2}) with the same value. Indeed $(c_i, c_{i'})$ is actually assimilated to (c_{ref1}, c_{ref2}) with score $W_i W_{i'}$. Usually 3 or 4 reference colors are enough to fairly describe an object, consequently memory saving is very significant. In addition Bhattacharyya distances computation coming afterwards is greatly simplified. One should also add a reject class, that will take into account any color too far from any reference color. Finally the size of the matrix is reduced to $(N_k + 1) \times (N_k + 1)$.
Eventually, one can also wonder about how many matrices should be computed. As seen in section II-A, the norm of vector $\vec{d}$ is usually set to 1, and its orientation is set to one of these four : $\left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}\right\}$. This helps to cope with object rotations. Instead of calculating four matrices, one for every orientation, we propose to calculate only one, but to compare it to 4 matrices, a priori learnt from the objet. The shortest distance is kept. Comparing two matrices being far less costly than building a matrix, overall computation time is thereby reduced. In the fields of car tracking this rotation invariance property is often unneeded, so the reference set can also be limited to only one matrix.

IV. RESULTS AND DISCUSSIONS

Unlike many authors our new feature will be judged upon tracking efficiency, not upon classification efficiency. Along with tracking purposes, this feature was designed in order to contain color-textural information even when slight illumination changes occur, as it is always the case in natural scenes. Two criteria will be examined : tracking precision and tracking robustness. The tests performed in this section have all been carried out using the particle filter described in the previous section. Similar settings have been used for all tests, notably the number of particles was set to 300, $\sigma_1 = 10$, $\sigma_2 = 10$, $\sigma_3 = 1$ and $\sigma_4 = 1$. For such settings and an average subwindow size of 15×15, one image was processed in 0.59 second. The algorithm was run on a laptop using a 2.0 Ghz intel Core Duo CPU.

A. Tracking precision

It is always difficult to objectively measure the precision of a tracking procedure. The best way to build some opinion about it is still the human eye. On figure 2 the particle filter was used with, Gabor features, classical cooccurrence matrices, and weighted cooccurrence matrices (WCCM) with three different kernels. Gabor features are often said to be the best texture characterization method [13]. Comparing methods through tracking efficiency is however not so usual. The driving sequence used for the test has a resolution of 240x320 pixels and is 486 frames long, approximately corresponding to 31 seconds. However only the most part of the sequence, during which appearance of the tracked car mainly changes, is displayed on figure 2.
According to this figure, Gabor features can produce reliable information about the car location for some seconds, but when changes on the car appearance are getting coarser, it loses track. Color cooccurrence method does perform some tracking, which proves that cooccurrences contain reliable information. However this tracking is very imprecise. As you may see, only part of the car rear view and its shadow is included in the bounding box. This kind of tracking quality does not match computer vision goals. When looking at tracking results from WCCM methods, it is clear that the tracking efficiency is enhanced, because the car is still globally contained by the bounding box. Concerning the different kernels, it appears that Gaussian and Epanechnikov ones both give very satisfying results, and that rational kernel is a little less accurate at the end of the sequence. Note that the filter has successfully come through slight illuminations changes, scale changes and view changes.

B. Tracking robustness

Looking at figure 2, one may say that robustness is more or less the same for the three different kernels used in the weighted cooccurrence matrices computation. To further discuss this hypothesis, we propose the following figure 3. On this figure, the evolution of the minimal distance measured over all particles is displayed for each kernel. We have divided these distances by the distance measured on the second image, because in this image the rendering of the car is very close to our model learnt on the first image. It appears that the rational kernel curve is globally under that of the Gaussian kernel, which is itself under the Epanechnikov one. Having a curve with low values means that the model is still close to the measurements made at each time of the

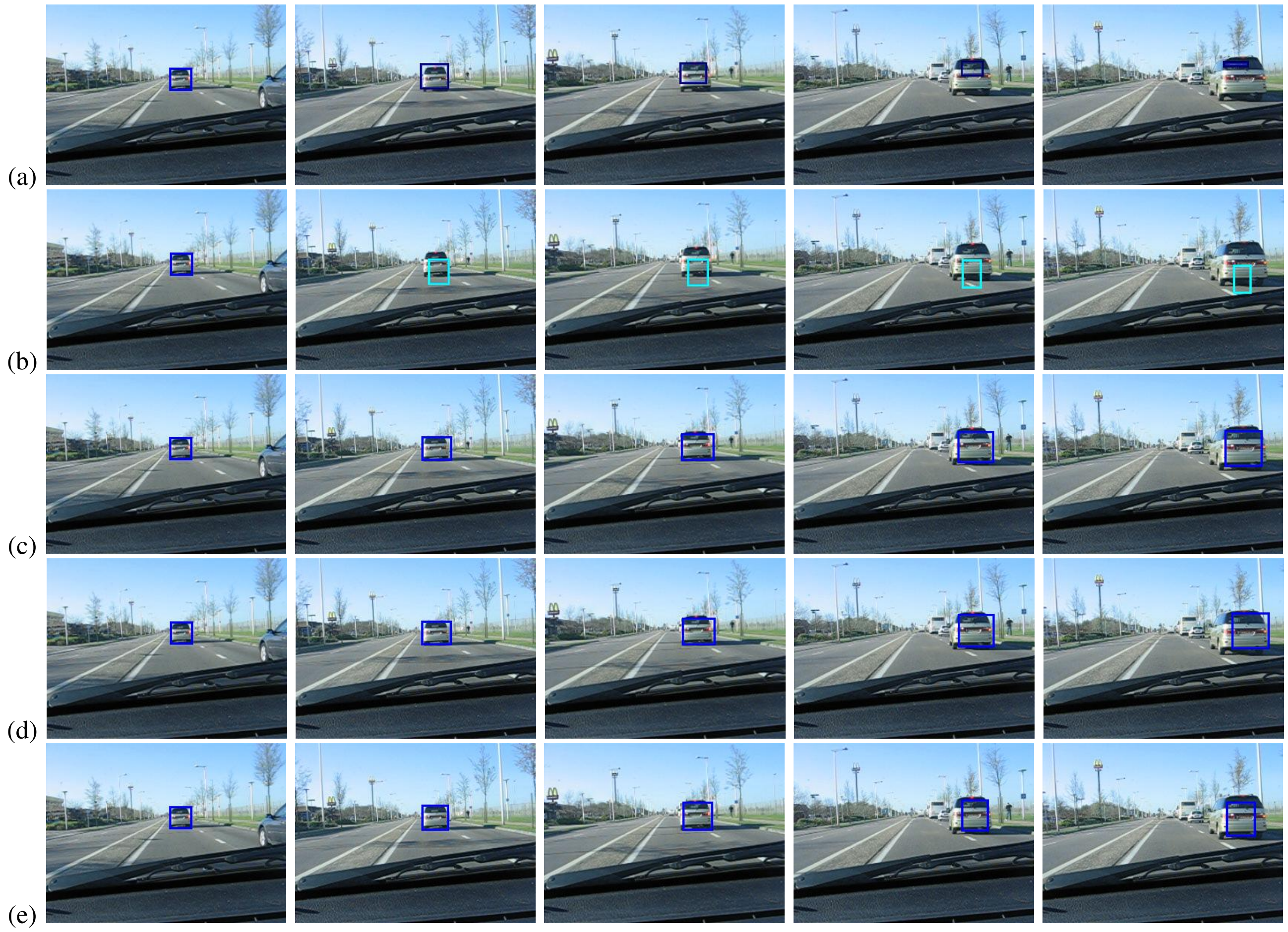

Fig. 2. (a) : tracking with Gabor features, (b) : tracking with classical cooccurrence, (c) : tracking with epanechnikov WCCM, (d) : tracking with Gaussian WCCM, (e) : tracking with rational WCCM

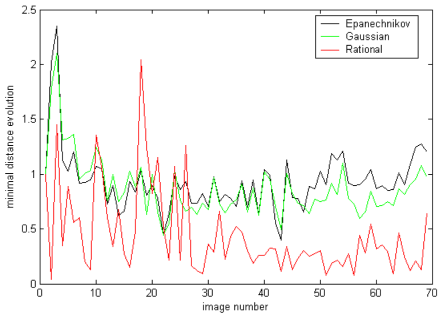

Fig. 3. Minimal distance evolution for different kernels. The distances are computed between the model and WCCM extracted from subwindows sampled by the particle filter.

sequence, in other words it means that the model is robust. Consequently, rational kernel turns out to be more robust than the two others. This observation can be explained by the remark made in section II. Indeed rational kernel takes into account more colors than the others, so it can more easily detect something close to the model, but it may also be misleading. Note that the variance of the rational kernel curve is obviously higher, which may be related to its lack of precision.

A second experiment confirms these results. From an image containing a front view of a car, a sequence was constructed using successive gamma functions for enlightening the first image. On top of figure 4, the extent of this artificial enlightening can be observed. The three different kernel cooccurences were tested on this new sequence, with same settings.

As figure 4 shows it, track is by far maintained longer with rational kernel. Even if this result is very good, we want to stress that such a coarse enlightening cannot be handled by our feature in the general case. In this example, the enlightening is homogeneous on the whole image, there is no motion, and the car possesses color properties that enables slight similarity to give relevant information about the car location. However it does prove the efficiency of our approach. Such artificial homogeneous enlightening is almost never observed on natural conditions, usually enlightening can be said to be locally homogeneous. Consequently this enlightening may be harder to cope with than natural ones, because more information is lost.

The tests performed in this section have brought to light a

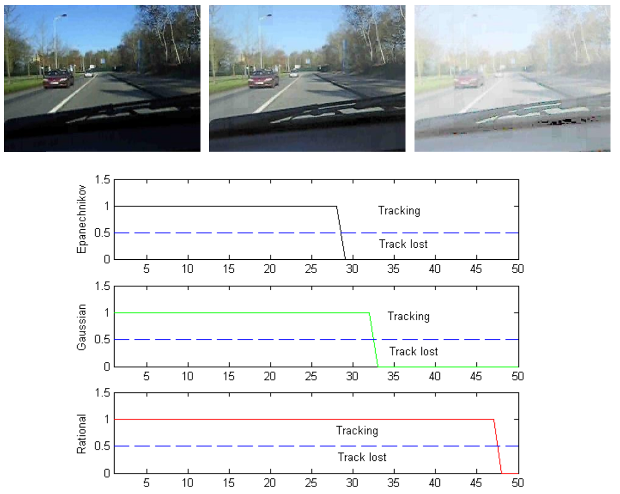

Fig. 4. Tracking robustness to illumination changes

paradox between robustness and precision. Regarding our experiments, we recommend the use of Gaussian kernel, which is a good compromise between the two others.

V. CONCLUSIONS

In this article a new color-texture feature tuned for tracking was introduced. It is built upon the color cooccurrence matrices. If an object has to be modeled by this method, it is possible to choose a few representative colors of the object. Thus, we propose to weight the vote of colors in the occurrence increment, depending on whether or not a color is close to some of the representative chosen ones, hence the name of weighted color cooccurrences for this new feature. The weighting is performed thanks to a kernel. Three kernels have been examined : rational, Gaussian and Epanechnikov. The experiment proved that weighted cooccurrences outperform classical cooccurrences or Gabor features in a car tracking context. In addition it appears that Epanechnikov and Gaussian kernels are precise, but rational kernel is more robust. As a consequence a compromise between robustness and precision must be reached.

Thanks to weighted cooccurrence matrices, a reliable color-texture source of information is available for tracking purposes. Yet color and texture are not the only cues that can be processed within an image. In future works, we intend to propose data fusion between our new feature and other features extracted from shape or movement analysis techniques. Thanks to this fusion, we hope to be able to analyze scenes at objects level, not only at pixels level, so as to handle more disrupting events for tracking, notably occlusions, clutters or pose changes.

REFERENCES

[1] E. Arnaud and E. Mmin. Optimal importance sampling for tracking in image sequences : application to point tracking. In *ECCV 2004*, 2004.

[2] M.S. Arumpalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2) :174–188, 2002.

[3] P. Chang and J. Krumm. Object recognition with color cooccurence histograms. In *IEEE Conf. on Computer Vision and Pattern Recognition*, Fort Collins, 1999.

[4] L. Cinque, S. Levialdi, A. Pellicano, and KA Olsen. Color-based image retrieval using spatial chromatic histograms. *IEEE Multimadia Systems*, 2 :969–973, 1999.

[5] D. Comaniciu, V. Ramesh, and P. Meer. Real time tracking of non rigid objects using mean shift. In *Comp. Vis. And Patten recognition conf.*, pages 142–149, Hilton Head Island, 2000.

[6] A. Doucet. On sequential simulation-based methods for bayesian filtering. *Technical report, Univ. of Cambridge*, 310, 1998.

[7] N.J. Gordon, A.F.M. Smith, and D.J. Saldmond. A novel approach to nonlinear/non-gaussian bayesian state estimation. *IEEE proceedings-F*, 40(2) :107–113, 1993.

[8] R. M. Haralick, K. Shanmugan, and I. Dinstein. Textural features for image classification. *IEEE Trans. on Systems, Man and Cybernetics*, 3(6) :610–621, 1973.

[9] C.R. Jung. Unsupervised multiscale segmentation of color images. *Pattern Recognition Letters, art. in press*, 2006.

[10] N. Shephard M. K. Pitt. Filtering via simulation : Auxiliary particle filters. *Journal of the American Statistical Association*, pages 590–599, 1999.

[11] C. Palm. Color texture classification by integrative co-occurrence matrices. *Pattern Recognition*, 37 :965–976, 2003.

[12] G. Paschos. Fast color texture recognition using chromacity moments. *Pattern Recognition Letters*, 21 :837–841, 2000.

[13] A. Drimbarean P.F. Whelan. Experiments in colour textures analysis. *Pattern Recognition Letters*, 22 :1161–1167, 2001.

[14] F. Suard, V. Guigue, A. Rakotomamonji, and A. Bensrhair. Pedestrian detection using stereo-vision and graph kernels. In *Intelligent Vehicules Symposium, Las Vegas, USA*, pages 267–272, June 2005.

[15] P. Torma and C. Szepesvari. Enhancing particle filters using local likelihood sampling. In *ECCV 2004*, pages 16–27, 2004.