Robust Adaptive Patch-based Object Tracking using Weighted Vector Median Filters

Leandro Dihl
PUC-RS/FACIN
leandro.dihl@acad.pucrs.br

Cláudio R. Jung
UFRGS - Institute of Informatics
crjung@inf.ufrgs.br

José Bins
UNIPAMPA at Alegrete
josebins@unipampa.edu.br

***Abstract*—This paper presents a novel patch-based approach for object tracking robust to partial and short-time total occlusions. Initially, the original template is divided into rectangular subregions (patches), and each patch is tracked independently. The displacement of the whole template is obtained using a weighted vector median filter that combines the displacement of each patch and also a predicted displacement computed based on the previous frames. An updating scheme is also applied to cope with appearance changes of the template. Experimental results indicate that the proposed scheme is robust to partial and short-time total occlusions, presenting a good compromise between accuracy and execution time when compared to other competitive approaches.**

***Keywords*-Object tracking; Occlusion; Multiple patches; Weighted Vector Median Filter; Bhattacharyya Distance**

I. Introduction

Nowadays, with the widespread use of cameras in our society, the need for automated video analysis has become imperative. This need has greatly increased the interest in object tracking algorithms, which main goal is to identify objects (or parts of objects) in a succession of frames. The object to be tracked is usually referred to as the *target*.

There are several different approaches to address the problem of object tracking relying on a variety of mathematical tools. However, there are still several factors that influence the performance of tracking algorithms, such as noise, complex object motion, occlusions, varying scene illumination, appearance changes and real-time requirements. In fact, the large number of papers published on this subject indicates that the problem is still open.

This work[1] presents a novel patch-based approach for object tracking. The initial template is split into smaller non-overlapping patches, which are tracked individually. The results of tracked patches are combined in a robust way to determine the displacement of the whole template, such that partial occlusions can be handled. Moreover, a predictor scheme is included in the tracking process, so that, in some cases, even complete occlusions can be handled. Finally, a fast updating scheme is applied to cope with appearance changes.

The remainder of this paper is organized as follows. Section II presents some related work on region-based object tracking, and Section III describes the proposed approach. Some experimental results are provided in Section IV, and a discussion of the proposed method is given in Section V. Finally, the conclusions are drawn in Section VI.

[1]Author Claudio Jung would like to thank CNPq for partially financing this work.

II. Related work

There are several approaches for object tracking in video sequences, and region-based techniques try to match the same region in adjacent frames, using a variety of matching functions. A classical approach is the KLT (Kanade-Lucas-Tomasi) tracker [1], which is based on minimizing the squared error of a (usually small) window centered at corner-like features. The minimization procedure is performed iteratively, being very fast, but facing problems in the presence of occlusions within the window.

Another popular approach for region-based object tracking is histogram matching. In [2], the authors explored the principle of color histogram distance within a probabilistic framework. In their approach, particle filtering was used to handle clutter in the background and occlusions. Comaniciu et al. [3] developed a kernel-based tracking scheme, where each region is characterized by its histogram, and masked with an isotropic kernel that assign smaller weights to pixels far from the center of the region. Similar templates in adjacent frames are found through an iterative scheme based on the Bhattacharyya coefficient. This approach is very fast and can deal with partial occlusions on the peripheral regions of the target, but tends to fail when the occlusion gets closer to the center of the target (where the weights for the histogram are larger). An important issue is that object kernels should have a certain overlap in the consecutive frames, despite recent efforts to improve convergence issues [4], [5]. Lerdsudwichai and collaborators [6] proposed a face tracking algorithm using the Bhattacharyya coefficient as a similarity measure for template matching, where each template is represented by color histograms (*CbCr* components in the *YCbCr* color model) modulated by Epachinekov kernels. The value of the similarity metric is used to detect total occlusions explicitly, and the color of the shirts are used to recover individual faces after the occlusion. Porikli [7] proposed a fast way to extract multidimensional histograms based on integral images, allowing exhaustive search for object tracking based on histogram matching with low computational cost. Marimon and Ebrahimi [8] combined gradient orientation histogram matching and template matching through Normalized Cross

Correlation, aiming to estimate rotation along with region matching. Porikli et al. [9] used the covariance matrix as a parametric representation of the target, and adopted a distance metric for covariance matrices based on generalized eigenvalues to compute region similarity.

Multiple patches have been adopted to overcome this problem, since some patches may present tracking errors when analyzed individually, but the non-occluded patches may lead to robust template tracking. Hager et al. [10] proposed a new iterative scheme for matching kernel-modulated histograms, and introduced objective functions optimizing more elaborate parametric motion models based on multiple spatially distributed kernels. Adam et al. [11] proposed a fragments-based tracking algorithm based on histogram matching using the Earth's Mover Distance (EMD). In their approach, different template fragments are used to build a single robust distance map, where occluded fragments present a smaller weight. Their approach presents good tracking results in substantial partial occlusions, but fails as the number of occluded patches increases. The approach in [12] employs the same basic idea of covariance matching as [9], but using multiple overlapping templates to cope with partial occlusions, and focused mainly on object recognition. Zhu et al. [13] dealt with occlusions by dividing the object in smaller blocks, and explored the similarity in local color and texture features for each of the blocks. Despite the good results, the approach in [13] uses a background subtraction algorithm, which limits its application to static cameras. Liu and colleagues [14] use a local sparse representation to model the appearance of target patches, and a sparse coding histogram is used to represent the basis distribution of the target. Despite the promising results, computational time was not informed by the authors.

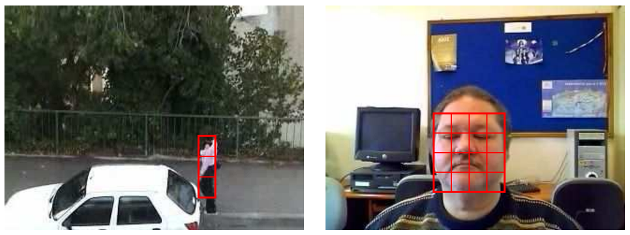
Although the use of multiple patches tends to present more robust tracking results, the presence of occlusions (particularly total occlusions) is still a challenge. This work presents a patch-based tracking approach, where each patch is tracked individually, and the individual displacement vectors are combined in a robust manner to obtain the overall displacement of the object. The proposed method is described next.

III. The Proposed Model

A. Patch-Based Templates

The use of multiple patches presents additional information for object tracking, that can be used to resolve ambiguities, model complex motion, and improve robustness w.r.t. occlusions. Clearly, this selection is highly dependent on the dissimilarity measure used to compare the patches. The ideal patch selection is a complex task, and it is out of the scope of this paper. In fact, our work follows the idea of [11] regarding patch selection. The original rectangular template, representing the region to be tracked, is split into a grid of $n \times m$ adjacent rectangular patches. Although this subdivision may produce ambiguous patches presenting erroneous tracking results, the remaining patches that present a correct match help to compute the displacement of the whole template in a robust manner. Also, a non-uniform patch distribution could bias the tracking result in the presence of partial occlusions, particularly when portions of the template containing a higher density of patches are occluded (so that just a few non-occluded patches would remain).

As it will become clear in Section III-B, each patch is represented through a mean vector and a covariance matrix. Hence, the selected patches must be sufficiently large to provide reliable estimates of these statistical parameters. Given the minimum desired side for each patch n_p, the user-selected target is divided uniformly into squares which sizes are as close to n_p as possible. Figure 1 illustrates the selection of patches for some video sequences analyzed later in this work, where $n_p = 20$ was experimentally defined as the default desired patch side. The outer (thicker) rectangle is the manually inserted template, and the smaller squares are the obtained patches.



(a) *Woman* (b) *Face1*

Fig. 1. Automatic selection of patches from the user-defined rectangular template.

B. Patch Matching

There are several methods for template matching based on pixel statistics, such as direct histogram matching using different similarity measures [8], [11], kernel-weighted histograms [3], or comparing statistical parameters that describe each region, such as the covariance tracking algorithm proposed in [9].

One problem of histogram-based methods is the rapid growth in the computational complexity as the dimension of the feature space increases: if N_b bins are used to represent each of the d dimensions, the histogram requires N_b^d bins for a full representation. For example, only intensity values are used in the histogram-based tracking algorithm proposed in [11], since the authors claim that using 3 color channels would lead to an expensive algorithm. On the other hand, statistical feature descriptors usually require a small amount of memory and matching metrics which are cheaper to implement, but the choice of the features and the statistical descriptors is still a challenge. For example, the tracking algorithm described in [9] uses a compact representation of each region (the covariance matrix, requiring $d(d+1)/2$ parameters) to perform region matching. However, it is clear that distinct regions may have the same covariance matrix, leading to possibly erroneous matching.

The definition of the dissimilarity measure for object matching is also a disputed issue. Rubner and collaborators [15] performed an empirical comparison of several matching algorithms (including the Bhattacharyya distance, Kullback-Leibler divergence and EMD, among others) focused on color and texture features, and concluded that "there is no measure with best overall performance, but the selection depends on the specific task".

For some parametric models, there are closed-form expressions for classical measures for comparing two probability density functions (such as the Bhattacharyya distance) that involve only the statistical parameters of both distributions. If the estimation of these parameters is cheap (in the computational point of view), such closed-form expressions are very fast to evaluate.

One model that can be computed very fast and that leads to a closed-form expression for the Bhattacharyya distance is the multivariate Gaussian distribution. This distribution is characterized by the sample mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{C}$, which can be computed efficiently in any rectangular region using the integral representation described in [12]. In fact, the cost for computing an integral image in a rectangular region with dimensions $H \times W$ using feature vectors of dimension d is $\mathcal{O}(HWd^2)$, and the cost to estimate $\boldsymbol{\mu}$ and $\boldsymbol{C}$ within any rectangular subregion of arbitrary size is $\mathcal{O}(d^2)$.

Given two Gaussian distributions with parameters $\boldsymbol{\mu}_1$, $\boldsymbol{C}_1$ and $\boldsymbol{\mu}_2$, $\boldsymbol{C}_2$, the Bhattacharyya distance between them is given by

$$\begin{aligned} B &= \frac{1}{8}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \left[\frac{\boldsymbol{C}_1 + \boldsymbol{C}_2}{2}\right]^{-1} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2) + \\ &+ \frac{1}{2}\ln\left(\frac{|(\boldsymbol{C}_1 + \boldsymbol{C}_2)/2|}{\sqrt{|\boldsymbol{C}_1||\boldsymbol{C}_2|}}\right). \end{aligned} \quad (1)$$

The matrix inversion is the most expensive operation when evaluating the dissimilarity, and it is traditionally computed with $\mathcal{O}(d^3)$ operations. Also, if the d features are uncorrelated, the covariance matrix is diagonal, and the cost to evaluate Equation (1) reduces to $\mathcal{O}(d)$.

One problem with the Gaussian assumption is that, in general, the distribution of features vectors (e.g. *RBG* color channels) is not even unimodal for most objects. In people tracking, for instance, the person may have a shirt in one color and the pants in another, leading to a bimodal distribution. Other models (such as a mixture of Gaussians) would be more adequate, but at a greater computational cost. In our approach, since the region of interest is split into smaller patches, the distribution of feature vectors within each patch is more likely to be roughly homogeneous, and the deviation from the normal may not be very large. A possible drawback of using patches is that large homogeneous regions in the template could accommodate different similar patches, which could lead to tracking errors for these patches. However, the overall displacement of the whole template is computed using a robust weighted average of individual displacements, so that a few patches with wrong results are overwhelmed by the remaining ones. In fact, the experimental results shown in Section IV indicate the robustness of the proposed technique.

To obtain the individual displacement vector $\mathbf{v}_i$ for each patch i, a search region with dimension $S_n \times S_m$ is placed at the center of the patch in the current frame. The Bhattacharyya distance between the patch and every candidate in the search region is computed exhaustively, and the selected patch is the one presenting the smallest distance.

C. Motion Prediction

Motion prediction can be very useful to reduce the search area or to eliminate spurious movements under heavy occlusion. There are two important issues with motion prediction: how to do it fast, and how to use the information in a easy and coherent way. In this work, we adopted the Double Exponential Smoothing technique proposed in [16] for position prediction, which is very fast and has prediction performance equivalent to Kalman filters.

Given a temporal series of vectors $\boldsymbol{v}_t$, the prediction at time $t + \tau$ is given by

$$\boldsymbol{v}_{t+\tau} = \left(2 + \frac{\alpha\tau}{1-\alpha}\right)\boldsymbol{Sv}_t - \left(1 + \frac{\alpha\tau}{1-\alpha}\right)\boldsymbol{Sv}_t^{[2]}, \quad (2)$$

where $\boldsymbol{Sv}_t$ and $\boldsymbol{Sv}_t^{[2]}$ are auxiliary variables computed through

$$\boldsymbol{Sv}_t = \alpha\boldsymbol{v}_t + (1-\alpha)\boldsymbol{Sv}_{t-1}, \quad (3)$$

$$\boldsymbol{Sv}_t^{[2]} = \alpha\boldsymbol{Sv}_t + (1-\alpha)\boldsymbol{Sv}_{t-1}^{[2]}. \quad (4)$$

Here, α is the degree of the exponential decay (smaller values for α produce smoother predictions). In the proposed approach, we used $\alpha = 0.1$ to get a smooth prediction, and $\tau = 1$ (to get a prediction at each frame) to generate a predicted vector $\boldsymbol{v}_p = \boldsymbol{v}_{t+1}$.

D. Combining Patch Information and Motion Prediction

The matching measure and the motion prediction described previously generate a set of N_p displacement vectors $\boldsymbol{v}_i$ (where N_p is the number of patches) and one predicted displacement vector $\boldsymbol{v}_p$, in a total of $N_p + 1$ individual motion information. For sakes of simplicity, we will define $\boldsymbol{v}_{N_p+1} = \boldsymbol{v}_p$.

For translational only movements, all these vectors should be similar. However, partial occlusions, patches on uniform regions and illumination changes may corrupt the displacement vector $\boldsymbol{v}_j$ for one or more patches. Computing the mean displacement vector would be a naïve approach, since the mean can be significantly affected by a single outlier. A more adequate approach, commonly used in color image denoising [17], is the use of Weighted Vector Median Filters (WVMF), that implicitly account for outlier rejection. In the original formulation of the WVMF [17], the first step consists of computing the distance from each vector to all others:

$$D_j = D(\boldsymbol{v}_j) = \sum_{i=1}^{N_p+1} \|\boldsymbol{v}_j - \boldsymbol{v}_i\|, \quad j = 1, ..., N_p + 1, \quad (5)$$

where $N_p + 1$ is the total number of vectors, and $\|\cdot\|$ is a vector norm (in this work, we employed the L_1 norm). The filtered vector $\boldsymbol{v}_f$ is then defined according to

$$\boldsymbol{v}_f = \frac{\sum_{i=1}^{N_p+1} w_i \boldsymbol{v}_i}{\sum_{i=1}^{N_p+1} w_i}, \tag{6}$$

where $w_i = f(D_i)$, and f is a nonnegative monotonically decreasing function (so that vectors that are farther from the median carry less weight).

As it can be observed, the weights w_i in the original WVMF formulation [17] include only geometrical distances between pairs of vectors. In this paper, we propose a modification of the weights w_i by also including the matching error (i.e. the Bhattacharyya distance) B_i of each patch according to Equation (1) in the filtering process, so that patches with smaller matching errors carry more weight. More precisely, the proposed weights for the WVMF are given by

$$w_i = g(D_i, B_i), \tag{7}$$

where $g(x, y)$ is a nonnegative monotonically decreasing function when considering the variables x and y individually, i.e., $\partial_x g(x, y) < 0$ and $\partial_y g(x, y) < 0$, $\forall x, y > 0$. With this choice for w_i, vectors that present smaller matching errors B_i and that are also geometrically consistent with the remaining vectors (i.e., present a smaller distance D_i) are prioritized in the weighted average.

One class of 1D functions that has shown good results for removing outliers in color image denoising [18] is $\exp(-x^r/\beta)$, where β and r are parameters that are chosen to give a good general purpose filter. Our choice for the 2D function g follows the same idea, and it is given by

$$g(x, y) = e^{-\left[(x/\beta)^2 + (y/\gamma)^2\right]}, \tag{8}$$

where β and γ control the decay of g as a function of x and y, respectively. Smaller values for β and γ prioritize displacement vectors $\boldsymbol{v}_i$ that present the smallest distance D_i and Bhattacharyya error B_i, respectively. As β and γ increase, the WVMF gets closer to the simple average filter, since all the weights w_i tend to be similar. In this work, we selected β adaptively through $\beta = \min_i D_i$, so that the decay of g is strong for displacement vectors far from the minimum value. We also propose to use $\gamma = 0.15$, since our experimental results indicated that non-occluded patches typically present Bhattacharyya errors around or below 0.15. The Bhattacharyya coefficient $b = e^{-B}$ relates to a bound for the classification error between two equiprobable classes [19], and it is used as a similarity measure in other works [3], [6]. Hence, $B = 0.15$ leads to $b \approx 0.86$, which is coherent with the similarity metrics reported in [6].

It should be noticed that there is no matching error B_{N_p+1} associated to the predicted motion vector $\boldsymbol{v}_{N_p+1}$, which is required in Equation (7). If a small value is assigned to B_{N_p+1}, the predicted displacement will carry more weight in the WVMF, and the opposite happens for larger B_{N_p+1} values. The first situation is adequate during total occlusions, since the patches do not provide reliable information. On the other hand, the second situation is suited for normal tracking conditions, where the information provided by the patches should dominate. In our approach, the value of B_{N_p+1} is computed adaptively in time, based on the patch errors in the previous frames.

For each frame t, there are $N_p + 1$ Bhattacharyya distances $B_i(t)$ related to the N_p patches and the predicted vector[2] $\boldsymbol{v}_{N_p+1}$. The representative error $B(t)$ for whole template at frame t is retrieved from the individual displacement information that is most coherent with the displacement of the whole template, i.e.

$$B(t) = B_j(t), \quad \text{where} \quad j = \underset{i}{\operatorname{argmin}} \|\boldsymbol{v}_f - \boldsymbol{v}_i\|. \tag{9}$$

In normal tracking conditions, $B(t)$ is expected to represent the Bhattacharyya distance of an existing patch. On the other hand, during total occlusions, the patches are expected to produce spurious displacement vectors (probably with large matching errors), and $B(t)$ tends to represent the Bhattacharyya distance of the predicted vector at the previous frame.

Finally, the selected error B_{N_p+1} for the predicted vector $\boldsymbol{v}_{N_p+1}$ is given by

$$B_{N_p+1} = \underset{k \in \{t-T_p, \ldots, t\}}{\operatorname{median}} B(k), \tag{10}$$

where T_p is the temporal window (in this work, we used $T_p = 30$ for videos acquired at 30 FPS, so that the median error in the past second is retrieved). Again, the rationale for this choice is that when the object starts being occluded, B_{N_p+1} should retrieve the matching errors of correctly matched patches in the previous frames, tending to present smaller errors than those of currently tracked patches (which are under occlusion). Hence, the prediction vector tends to carry more weight during occlusions.

It should also be noticed that there is only one predicted displacement vector $\boldsymbol{v}_{N_p+1}$, and N_p displacement vectors generated from the patches. If N_p is large, the overall displacement of the template tends to be dominated by the patch displacements $\boldsymbol{v}_i$, $i = 1, \ldots, N_p$, even if B_{N_p+1} is small. To cope with this issue, we also introduced a compatibility factor $0 \le c \le 1$ to re-weight $\boldsymbol{v}_{N_p+1}$ as a function of N_p. More exactly, the weight w_{N_p+1} in Equation (6) is re-computed through

$$w_{N_p+1} = c N_p g(D_{N_p+1}, B_{N_p+1}). \tag{11}$$

Smaller values for c decrease the weight of $\boldsymbol{v}_{N_p+1}$, and the opposite happens for larger values of c. In this work, we used $c = 0.5$ in all experiments.

An example of the procedure for combining the individual displacement vectors for each patch to obtain the overall displacement of the whole template is illustrated in Fig. 2. Fig. 2(a) shows the initial template (outer red rectangle) and

[2]Except for the first frame, where $B_{N_p+1}(t)$ is not defined.

the individual patches (smaller subrectangles). The vectors in blue indicate the individual displacement vectors for each patch, and the green vector at the center refers to the global displacement of the template. The displaced template in the subsequent frame is illustrated in Fig. 2(b). It is interesting to notice that the bottom patches were occluded from one frame to the other, and erroneous displacement vectors were created. Also, some unoccluded patches presented misleading displacements. However, the WVMF discarded the influence of these vectors, leading to a correct displacement for the whole template.

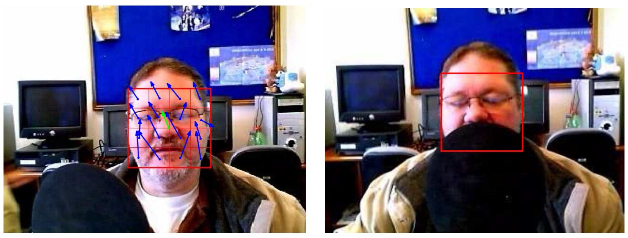

(a) (b)

Fig. 2. Example of the procedure adopted to obtain the displacement of the whole template based on the displacement of each individual patch through WVMFs.

E. Model Update

To cope with object changes (appearance, illumination, etc.), the model must be updated to improve tracking results. A simple and direct approach would be to compute the covariance using tracked templates in different time steps, as noticed by Porikli et al. [9]. However, as indicated by the same authors, the cost to compute the covariance matrix directly using T templates with dimension $M_T \times N_T$ is $\mathcal{O}\left(N_T M_T T d^2\right)$, which is very expensive and requires great amounts of memory. In fact, they proposed an updating scheme based on Lie Algebras to overcome this problem.

In this work, we use a fast and memory-efficient updating scheme for both the mean vector and covariance matrix, based on an iterative recomputation of these statistical descriptors. Let $\boldsymbol{\mu}_1$ and $\boldsymbol{C}_1$ be the mean and covariance of the model frame $t-1$, and let $\boldsymbol{\mu}_2$ and $\boldsymbol{C}_2$ denote the same parameters for current frame t. The updated mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{C}$ are given by

$$\boldsymbol{C} = (1-w)\left(\boldsymbol{C}_1 + \boldsymbol{\mu}_1\boldsymbol{\mu}_1^T\right) + w\left(\boldsymbol{C}_2 + \boldsymbol{\mu}_2\boldsymbol{\mu}_2^T\right) - \boldsymbol{\mu}\boldsymbol{\mu}^T \quad (12)$$

$$\boldsymbol{\mu} = (1-w)\boldsymbol{\mu}_1 + w\boldsymbol{\mu}_2, \quad (13)$$

where $0 < w < 1$ is the update rate of the model, such that $w \approx 1$ leads to a faster update, and $w \approx 0$ leads to a slower update of the model. If $\boldsymbol{\mu}_1$, $\boldsymbol{C}_1$ relate to a set S_1 containing N feature vectors, and $\boldsymbol{\mu}_2$, $\boldsymbol{C}_2$ relate to a set S_2 containing M feature vectors, it can be shown that the mean and covariance using all vectors in $S_1 \cup S_2$ can be obtained with Equations (12) and (13) using $w = M/(M+N)$ and $1-w = N/(M+N)$. Our experimental results indicated that $w = 0.1$ produced good results, and it was selected as a default value.

The computational cost of this procedure is $\mathcal{O}\left(d^2\right)$, and it does not depend on the dimensions of the template. Furthermore, as Equations (12) and (13) are applied recursively every frame, the resulting mean and covariance matrix embed information about all previous frames to which the updating rule is applied. The proposed scheme is also very efficient in terms of memory storage: only the previous descriptors ($\boldsymbol{\mu}_1$, $\boldsymbol{C}_1$) and the current descriptors ($\boldsymbol{\mu}_2$, $\boldsymbol{C}_2$) are used, and the samples used to compute the model in the previous frames are not required.

IV. Experimental Results

This Section presents some experimental results obtained with the proposed algorithm, called the Coherent Patch Displacement (CPD) Tracking algorithm. The experimental validation was performed qualitatively, by visual inspection of tracking results, and also quantitatively, by comparing the tracking errors produced by the proposed approach and by two state-of-the-art techniques, namely the MeanShift algorithm [3] and the FragTrack algorithm [11].

All the results presented in this Section were computed using C++ implementations of the algorithms (the code for FragTrack was kindly provided by Amit Adam), running on a PC computer with a Pentium Core 2 Duo 2.33GHz processor, 1GB RAM and windows XP operational system. To compare the techniques we used five different video sequences: *Woman*, *Caviar*, *Face1*, *Face2*, and *Person*. The *Woman* sequence is a video of a woman walking on a sidewalk, sometimes partially occluded by different cars (available for download at http://www.cs.technion.ac.il/~amita/fragtrack/fragtrack.htm). The *Caviar* sequence is one of the videos of the CAVIAR project (http://homepages.inf.ed.ac.uk/rbf/CAVIAR), and it consists of some people walking through a mall. *Face1* and *Face2* are two facial video sequences, containing head tilts, turns and occlusions. Finally, the *Person* video sequence illustrates a full body person moving behind several trees in an outdoor environment with illumination changes (from cloudy to partly sunny), shot with a moving camera. The *Woman* and *Caviar* video sequences are available with ground truth data, while the remaining sequences were manually ground truthed by our group. Tracking results for all video sequences, as well as the original versions of *Face1*, *Face2* and *Person* sequences with ground truth data can be accessed at http://www.inf.ufrgs.br/~crjung/cpd/cpd_videos.html.

For these five video sequences we computed the execution time and the tracking error (Euclidean distance between the actual position and the tracked position) for each frame. In all experiments, we used the same search region (30×30) for CPD Tracking and FragTrack (Meanshift does not require a search region). For FragTrack, we used 16 histograms bins (only luminance information), and the EMD distance as the histogram matching procedure. For CPD Tracking, we used the same set of 5 features for all sequences: the 3 color channels and the 2 components of the gradient computed from the luminance component. It is important to emphasize that, although CPD presents other tunable parameters (n_p, w, T_p,

γ, and c), the same default values described in the previous Section were used in all experiments, although, even better results could be achieved by fine tuning those parameters to each individual video sequence.

The plots in Fig. 3 show the tracking errors obtained with the algorithms for the video sequences. As it can be observed, Meanshift presents larger errors for all videos, mostly due to occlusions. Under partial occlusions (which happen frequently in the *Woman* and *Caviar* sequences), FragTrack and CPD Tracker present similar errors. However, under significant partial occlusions (as the final portion of the *Caviar* video) or total occlusions (as the regions marked with gray rectangles in the *Face1*, *Face2* and *Person* sequences), FragTrack gets lost, and it can only recover the target if it reappears within its search region. On the other hand, CPD Tracker can handle those situations efficiently. and in the first total occlusion of the *Person* sequence (around frame 230 in Fig. 3(e)). For these three video sequences, We also tried to run FragTrack with an expanded search region (70×70) to evaluate its ability to recover the target. The target was effectively recovered for the *Face1* sequences after the occlusion, but FragTrack failed to find the target after the second total occlusion of the *Person* sequence (around frame 330). In fact, the code for FragTrack (provided by the authors of [11]) halted for the *Person* sequence during the execution, when the template reached the boundary of the image after getting lost. This happened at frames 282 (30×30 search region) and 404 (70×70 search region). The same happened after the first occlusion of the *Face2* video sequence. Since there was no improvement when using the extended search region for this sequence, the error plot was not included in Fig. 3(d). It is also interesting to note that MeanShift may recover or not from total occlusions, depending on the color distributions of the target and its neighborhood at the time of the occlusion. If the iterative process that guides MeanShift pushes the template towards the target right after the total occlusion (as in *Face1* and *Face2* sequences), the target can be successfully recovered. On the other hand, if the template is pushed away from the real target (as in the *Person* sequence), the target can not be recovered.

A summary of the results is illustrated in Table I, that shows the mean error, maximum error and standard deviation for each technique and video sequence, as well as the average running time. As it can be observed, Meanshift presented the largest errors, followed by FragTrack and CPD Tracker. It is interesting to note that both FragTrack and CPD tracker achieved similar results in terms of accuracy when only partial occlusions appear (*Woman* and *Caviar* sequences), but CPD Tracker clearly outperforms FragTrack during total occlusions (the three other sequences). In terms of execution time, CPD was much faster than FragTrack: more than fifty times faster in the best case and more than twenty times faster in the worst case, considering the same search region 30×30. amplified when comparing CPD with the extended search size for FragTrack. The MeanShift tracker is the fastest of them all, mostly because it is based on an iterative search procedure instead of exhaustive search. However, its accuracy and robustness to occlusions (either partial or total) was far behind FragTrack and CPD.

TABLE I
TRACKING ERROR (IN PIXELS) AND EXECUTION TIME (IN SECONDS PER FRAME) FOR THE TRACKING METHODS: CPD TRACKER, FRAGTRACK AND MEANSHIFT. SMALLEST VALUES ARE DISPLAYED IN BOLD.

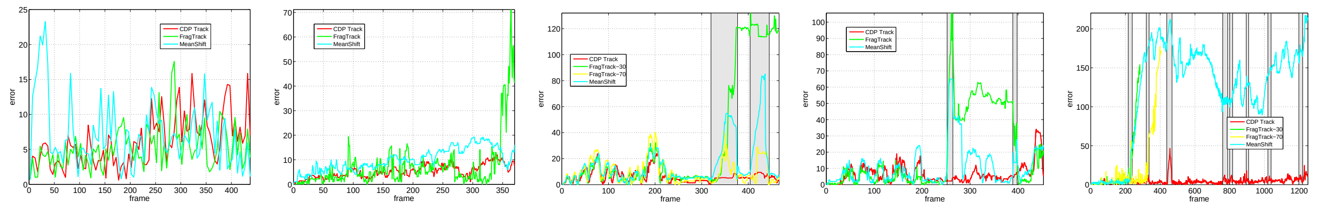
		CPD	FT-30	MS	FT-70
Woman	Average Error	6.03	**5.23**	7.20	–
	Max. Error	**15.86**	17.57	23.28	–
	Std. Deviation	3.65	**3.09**	4.75	–
	Avg. Time	0.055	2.892	**0.036**	–
Caviar	Average Error	**5.30**	7.33	10.05	–
	Max. Error	**12.04**	71.18	19.31	–
	Std. Deviation	**2.60**	10.38	4.56	–
	Avg. Time	0.050	2.323	**0.034**	–
Face1	Average Error	**6.44**	33.19	15.13	10.15
	Max. Error	**25.24**	132.02	85.15	41.34
	Std. Deviation	**4.90**	45.35	16.48	9.35
	Avg. Time	0.121	2.648	**0.032**	14.68
Face2	Average Error	**6.49**	19.94	10.95	–
	Max. Error	**34.13**	105.60	65.00	–
	Std. Deviation	**5.96**	23.75	12.06	–
	Avg. Time	0.061	2.819	**0.031**	–
Person	Average Error	**4.50**	20.04	120.65	24.53
	Max. Error	**46.57**	154.26	217.37	176.65
	Std. Deviation	**4.38**	38.63	63.34	43.60
	Avg. Time	0.049	2.204	**0.031**	7.53

Fig. 4 shows some frames of the five video sequences, along with the tracking template produced by each technique. In the first frame of each video, all techniques were initialized with the same template (so that only one template is visible). In frame 17 of the *Woman* sequence, when part of the woman's body is covered by the car, MeanShift has its template displaced, showing that it presents difficulties to deal with partial occlusions. On the other hand, both FragTrack and CPD are able to follow the woman correctly. The same happens in occlusion situations for the *Caviar* sequence. However, at the end of this sequence (see frame 366), the partial occlusion is substantial, and FragTrack misses the target. The other three sequences present situations of total occlusions. As it can be observed, CPD is usually able to estimate the target position under complete occlusions, while both Meanshift and FragTrack wrongly find the best match on the background (for instance, see frame 360 of the *Face1* sequence). In some cases (as in the *Face1*, *Face2* and *Person* sequence), Meanshift and/or FragTrack can not recover the target after the occlusion, and the matching template wanders around the image. Since Meanshift is based on an iterative procedure, it may not recover the target if it reappears sufficiently far from the current position of the template. The ability to recover the target for FragTrack is highly dependent on the search region, but the computational cost increases as the search region is expanded. In fact, CPD also presents the same drawback, but since it predicts the motion during occlusions, the search region can be kept relatively small.

V. DISCUSSION

A. Selection of Feature Vectors

It is important to note that several combinations of feature vectors can be used to build the model for each patch. Such



(a) *Woman* (b) *Caviar* (c) *Face1* (d) *Face2* (e) *Person*

Fig. 3. Tracking error for the analyzed techniques (MeanShift, FragTrack and CPD) for the five video sequences. Gray rectangles indicate portions of total occlusion.

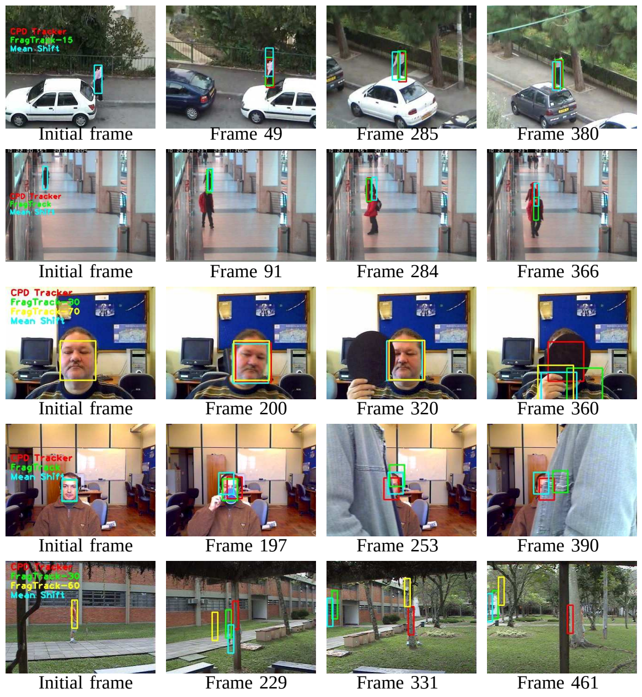


Fig. 4. Example frames for the five video sequences. Color rectangles indicate the matching template for the three tracking algorithms: CPD (in Red), FragTrack (in Green), and MeanShift (in Blue). Additionally, we included the results of FragTrack (in Yellow) with an extended search region (70 × 70) for some of the sequences.

features may be color (using different color spaces), gradient, second derivatives, textural information, depth information (when stereo cameras are employed), thermal, etc. Clearly, the best features are those that present a better separability between the chosen object and the neighborhood, what is very context-dependent.

We have tested the performance of our method using a variety of combinations involving different color spaces (such as *HSV* and normalized *RGB*), and concluded that *RGB* features plus gradient information presents good results in most of the cases (and no additional cost for color space transformation is required). In fact, just using *RGB* features is usually enough to achieve good tracking results, and the inclusion of gradient information appeared to improve the results under occlusions. Table II illustrates the tracking errors of CPD using only *RGB* features for the video sequences, along with execution times. As it can be observed, errors are typically a little larger than those obtained with *RGB* plus gradient, but running times are considerably lower (in most cases, even lower than MeanShift). A theoretical analysis of the computational cost is provided next.

TABLE II
TRACKING ERROR (IN PIXELS) AND EXECUTION TIME (IN SECONDS PER FRAME) FOR THE CPD TRACKER USING ONLY *RGB* FEATURES.

Woman		*Caviar*		*Face1*		*Face2*		*Person*	
Avg.	Max.	Avg.	Max.	Avg.	Max.	Avg.	Max.	Avg.	Max.
5.08	13.89	7.42	19.70	6.31	26.63	11.14	43.10	4.99	47.54
Std.	Time	Std.	Time	Std.	Time	Std.	Time	Std.	Time
3.09	0.029	4.67	0.031	5.19	0.049	7.80	0.031	5.20	0.018

B. Computational Cost

To analyze the computational cost of the proposed approach, let us consider that the initial template is split into N_p patches, and the patch statistics are computed using d-dimensional feature vectors. Also, let us consider a search region with dimension $S_n \times S_m$.

The cost for computing the integral image representation restricted to the search region is $\mathcal{O}(S_n S_m d^2)$, and the complexity for obtaining the mean and covariance of all N_p patches is $\mathcal{O}(N_p d^2)$. The cost to compute the distance between two patches using the Bhattacharyya distance is $\mathcal{O}(d^3)$[3], and the cost for an exhaustive comparison of all N_p patches within the $S_n \times S_m$ region is $\mathcal{O}(N_p S_n S_m d^3)$. The cost for obtaining the displacement of the whole template based on the individual patches using WVMFs is $\mathcal{O}(N_p^2)$, and the updating rule presents a cost of the order $\mathcal{O}(d^2)$. Hence, the total cost of the proposed procedure is $\mathcal{O}(S_n S_m d^2 + N_p d^2 + S_n S_m N_p d^3 + N_p^2 + d^2)$. In practice, $S_n S_m N_p d^3$ is by far the largest term, so that the complexity for a sequential implementation can be approximated as $\mathcal{O}(S_n S_m N_p d^3)$. It should be noted that, since each patch is tracked individually, parallel hardwares (such as multiple/multicore processors or Graphics Programmable Units) can be explored to further reduce running times.

C. Limitations

Although the proposed method performs usually well under partial and short-time total occlusions, there are some scenarios where it is prone to errors. For example, during longer-term occlusions, the updating rule would learn the statistics of the

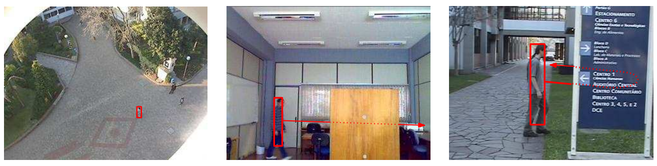
[3]However, as mentioned previously, there are algorithms that reduce the complexity to $\mathcal{O}(d^{2.376})$ [20].

occluding object. Also, if the target appearance is significantly changed during a total occlusion, it may not be recovered when it reappears from the occlusion.

Another issue regards the motion of the target during the occlusion. The motion prediction scheme utilizes the displacement vectors in the previous frames to estimate the future position of the target. If the target changes its motion during the occlusion (e.g. a person moving behind a wall stops during the occlusion), the predicted position would provide a wrong estimate.

It should also be noticed that the patches are tracked based on estimates of the mean vector and the covariance matrix. If smaller patches are used, the estimation of these parameters may be very sensitive to changes in a few pixels of the patches, which may lead to erroneous tracking results.

Examples of situations where the proposed approach may fail are illustrated in Fig. 5. In Fig. 5(a), the target is very small, and a subdivision into even smaller patches would lead to unreliable estimates of the mean vector and covariance matrix. In Fig. 5(b), the target gets occluded behind the wardrobe for a long time (dashed line), so that the template learns the statistics of the wardrobe and do not recover the target after the occlusion. Finally, Fig. 5(c) shows a target that changes completely its motion pattern during the occlusion, so that the predicted template would be far from the actual target after the occlusion.



(a) Small target (b) Long occlusion (c) Motion change

Fig. 5. Situations where the proposed approach may fail.

VI. Conclusions

This paper presented a robust patch-based approach for object tracking. The initial template is divided into disjoint rectangular subregions (patches), and each of these patches is tracked independently by minimizing the Bhattacharyya distance within a search region. The displacement vector of each patch is combined with a motion estimated vector using a WVMF, and the displacement of the whole template is obtained. Finally, a simple and efficient updating scheme is used to cope with appearance and illumination changes.

Experimental results indicated that the proposed method is robust with respect to partial and short-term total occlusions, and it can effectively adapt to appearance and illumination changes. Quantitative results indicated that the performance of the model is comparable to or better than competitive approaches [3], [9], [11], presenting the best compromise between tracking accuracy and execution time among all tested methods.

There are several avenues left for improving this work. For example, a comprehensive study on feature selection for region-based tracking could improve the results of the proposed algorithm. In fact, an ideal approach would consider adaptive online feature selection based on the characteristics of the object and its surroundings at each frame, similarly to the method described in [21]. Also, the use of templates with adaptive sizes could account for scale variations of the tracked object.

References

[1] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 593–600, 1994.

[2] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *European Conference on Computer Vision*, pp. 661–675, 2002.

[3] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.

[4] C. Shen, M. J. Brooks, and A. van den Hengel, "Fast global kernel density mode seeking with application to localisation and tracking," in *IEEE International Conference on Computer Vision*, pp. 1516–1523, 2005.

[5] C. Shen, M. J. Brooks, and A. van den Hengel, "Fast global kernel density mode seeking: applications to localisation and tracking," *IEEE Transactions on Image Processing*, vol. 16, no. 5, pp. 1457–1469, 2007.

[6] C. Lerdsudwichai, M. Abdel Mottaleb, and A. Ansari, "Tracking multiple people with recovery from partial and total occlusion," *Pattern Recognition*, vol. 38, pp. 1059–1070, July 2005.

[7] F. Porikli, "Integral histogram: A fast way to extract histograms in cartesian spaces," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 829–836, 2005.

[8] D. Marimon and T. Ebrahimi, "Orientation histogram-based matching for Region Tracking," in *Eighth International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2007)*, 2007.

[9] F. Porikli, O. Tuzel, and P. Meer, "Covariance tracking using model update based on lie algebra," in *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 728–735, 2006.

[10] G. D. Hager, M. Dewan, and C. V. Stewart, "Multiple kernel tracking with SSD," in *IEEE International Conference on Computer Vision*, vol. 1, pp. 790–797, June 2004.

[11] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Conference on Computer Vision and Pattern Recognition*, pp. 798–805, 2006.

[12] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *European Conference on Computer Vision*, vol. 2, pp. 589–600, 2006.

[13] L. Zhu, J. Zhou, and J. Song, "Tracking multiple objects through occlusion with online sampling and position estimation," *Pattern Recognition*, vol. 41, no. 8, pp. 2447–2460, 2008.

[14] B. Liu, J. Huang, C. Kulikowski, and L. Yang, "Robust tracking using local sparse appearance model and k-selection," in *In Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2011.

[15] Y. Rubner, J. Puzicha, C. Tomasi, and J. M. Buhmann, "Empirical evaluation of dissimilarity measures for color and texture," *Computer Vision and Image Understanding*, vol. 84, no. 1, pp. 25–43, 2001.

[16] J. J. LaViola, "Double exponential smoothing: an alternative to kalman filter-based predictive tracking," in *Proceedings of the Workshop on Virtual Environments*, pp. 199–206, 2003.

[17] J. Astola, P. Haavisto, and Y. Neuvos, "Vector median filters," *Proceedings of the IEEE*, vol. 78, pp. 678–689, 1990.

[18] M. Cree, "Observations on adaptive vector filters for noise reduction in color images," *IEEE Signal Processing Letters*, vol. 11, pp. 140–143, February 2004.

[19] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Wiley-Interscience Publication, 2000.

[20] D. Coppersmith and S. Winograd, "Matrix multiplication via arithmetic progressions," *Journal of Symbolic Computing*, vol. 9, no. 3, pp. 251–280, 1990.

[21] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Transactions on Pattern Analysis and Machince Intelligence*, vol. 27, no. 10, pp. 1631–1643, 2005.