Sample and Pixel Weighting Strategies for Robust Incremental Visual Tracking

Javier Cruz-Mota, Michel Bierlaire, and Jean-Philippe Thiran, *Senior Member, IEEE*

***Abstract*—In this paper, we introduce the incremental temporally weighted principal component analysis (ITWPCA) algorithm, based on singular value decomposition update, and the incremental temporally weighted visual tracking with spatial penalty (ITWVTSP) algorithm for robust visual tracking. ITWVTSP uses ITWPCA for computing incrementally a robust low dimensional subspace representation (model) of the tracked object. The robustness is based on the capacity of weighting the contribution of each single sample to the subspace generation to reduce the impact of bad quality samples, reducing the risk of model drift. Furthermore, ITWVTSP can exploit the *a priori* knowledge about important regions of a tracked object. This is done by penalizing the tracking error on some predefined regions of the tracked object, which increases the accuracy of tracking. Several tests are performed on several challenging video sequences, showing the robustness and accuracy of the proposed algorithm, as well as its superiority with respect to state-of-the-art techniques.**

***Index Terms*—Online learning, principal component analysis (PCA), visual tracking (VT).**

I. Introduction

VISUAL TRACKING (VT) is a core problem in many computer vision (CV) applications, such as human–computer interaction (HCI) [1]–[3], traffic monitoring [4], [5], video surveillance [6], [7], or augmented reality (AR) [8]. The main task of a tracking algorithm is to assign consistent labels to tracked objects along all the frames of a video sequence. Given a video sequence S

$$S = \{I_k | k \in K \subseteq \mathbb{N}\} \tag{1}$$

where k is a temporal index and I_k is a video frame, a tracking algorithm estimates a time series

$$x^{(j)} = \{x_k^{(j)} | k \in K \subseteq \mathbb{N}\} \tag{2}$$

for every tracked object $j \in J$, where J denotes the set of objects being tracked. Each element $x_k^{(j)}$ of the time series $x^{(j)}$ denotes the state of object j at time k, and defines its trajectory over time.

From a bottom-up point of view, a VT algorithm can be roughly defined by describing three main blocks [9], [10]: the feature extraction block, the object representation block, and the object localization block. A generic VT algorithm can be seen as the application of these three blocks according to the schematic representation in Fig. 1. Given a frame of a video sequence, the first block performs a feature extraction on its captured visual information. Feature extraction defines the space where the object of interest will be defined, such as colour, motion, edges, or interest points. The object localization block takes information from both the object representation model and the features, and estimates the new state of the object of interest. In general, localization is performed under the hypothesis of a smooth change of position, shape and appearance. Object localization algorithms can compute the target state analytically as the solution of an optimization problem where a cost function is minimized [11]–[15], or by evaluating simultaneously multiple candidate tracks (hypothesis) per object of interest per time step using particle filters (PF) [16]–[20]. PF validate these hypothesis against visual information and motion models. Their main advantage over analytical methods is their capacity to deal with multimodality and therefore with clutter.

All these object localization methods use the information supplied by the object representation model. This model contains information about the shape and/or the appearance of the object of interest. A wide variety of techniques are employed in the literature for computing the object representation model, such as principal component analysis (PCA) [18], [21], [22], mixtures of Gaussians [23], [24], histograms [25], [26], Bayesian approaches [27]–[29], boosting techniques [30], [31] or sparse representations [32], [33]. The critical point is that the appearance of the tracked object is continuously changing, and the model needs to be either built for dealing with these changes or to have the capacity of being adapted to them. In the first option, the changes have to be predicted and taken into account in the model estimation process, which is performed *a priori* or during an initialization period. In the second one, the model is constantly adapted to the tracked object with new data coming from the tracking, which keeps it permanently adapted to the object of interest and their current conditions. This second strategy is more effective in terms of adaptability since the type of change that the model has to handle does not need to be known beforehand. However, the

Manuscript received October 31, 2011; revised February 21, 2012 and May 9, 2012; accepted December 26, 2012. Date of publication February 26, 2013; date of current version May 1, 2013. This work was supported in part by the Swiss National Science Foundation under Grant 200021-117823. This paper was recommended by Associate Editor S. Yan.

J. Cruz-Mota and M. Bierlaire are with the Transport and Mobility Laboratory, École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland (e-mail: me@javiercruz.com; michel.bierlaire@epfl.ch).

J.-P. Thiran is with the Signal Processing Laboratory (LTS5), École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland (e-mail: jean-philippe.thiran@epfl.ch).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCSVT.2013.2249374

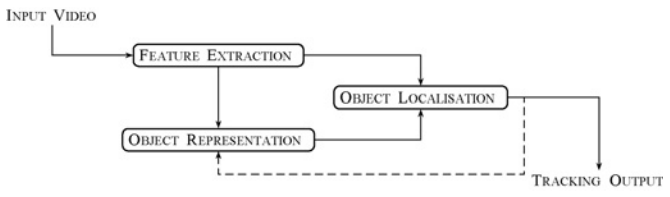


Fig. 1. Schematic view of a general VT algorithm.

adaptation procedure is very sensitive due to the possibility of corrupting the model with bad samples of the object of interest, causing a model drift and the consequent loss of track.

Some of the above-mentioned techniques can be used for computing a constantly adapted object representation model, while trying to avoid its corruption. However, one of the most well-known and simplest is probably PCA. PCA is a dimensionality reduction technique [34]. It consists of projecting the data onto the eigenvectors with biggest eigenvalues of the data covariance (or autocorrelation) matrix. In spite of its popularity and good performance, PCA presents two main problems: computational cost and sensitivity to outliers. The computational cost can be split by considering data incrementally [35]–[18]. This way, instead of computing a big PCA on a big data matrix, a PCA is performed on a small submatrix. This PCA is afterwards updated with new elements of the remaining dataset. Incremental procedures are also interesting when the whole dataset is not available at the beginning. In [36], the sequential computation of PCA is tackled by updating an existing PCA with the orthogonal components of the new data. Indeed, the process starts by computing, for the first block of data, its singular value decomposition (SVD), which is an efficient way of computing the principal components of a matrix. Then, for each new block of data, the update process is based on a QR factorization and a SVD of a small matrix. Their results, correctly combined, provide the principal components of the concatenation of the old and the new data matrices. Computation by blocks is considerably more efficient than updating the PCA for every new data sample, which makes methods based on [36] more efficient than those based on [35] (see [38] for comparison details).

With respect to the minimization of outliers' impact, two strategies arise in a PCA computation procedure. The first option is to discard samples that are supposed to be outliers. This forces to have a good outlier detection approach to do not discard good samples. For instance, in [39], a minimum volume ellipsoid is fitted to data to discard, in the PCA computation, the samples that are outside; and in [40] and [41], samples are discarded according to their projection in a subspace computed using a robust covariance estimation. The second option is to weigh the contribution of each value of the data matrix according to a measure of confidence. In these kinds of approaches, the PCA computation is still dependent on the outlier detection method, but its dependency is considerably weakened since all received data is considered. Kriegel *et al.* [42] use a weighted covariance matrix for computing the principal components. The weight values are computed using a distance function to clusters of points of the dataset. In [43], two kinds of weights are considered: temporal weights and spatial weights. Temporal weights adjust the contribution of each observation (a column in a data matrix), while spatial weights adjust the contribution of each variable (individual elements of each column). In [44], an incremental weighted PCA algorithm is introduced. The drawback of this algorithm is that it is based on the incremental PCA algorithm introduced in [35] and therefore updates the existing PCA for every single new sample. As commented before, this strategy is less efficient than updating the PCA with blocks of data.

Two main contributions are introduced in this paper. On the one hand, an incremental PCA algorithm with temporal weights is developed: the incremental temporally weighted principal component analysis (ITWPCA) algorithm. It is based on SVD update [36] and therefore can compute the incremental step using blocks of new data instead of individual samples, which makes the algorithm more computationally efficient than existing ones. On the other hand, a VT algorithm, based on a particle filter approach and the ITWPCA algorithm for object representation, is also developed: the incremental temporally weighted visual tracking with spatial penalty (ITWVTSP) algorithm. ITWVTSP computes an object representation model of the tracked object using ITWPCA on rectangular templates. The use of the ITWPCA algorithm for computing the PCA allows maintaining the object representation model constantly adapted to the tracked object, while reducing the impact of bad quality samples on the PCA. The last is achieved by computing a measure of the quality of every tracked sample, which modulates their contribution to the computed PCA. Furthermore, a strategy for spatial weighting of samples, directly on the particle weight computation, is also introduced. This spatial weighting allows assigning more importance to some predefined regions of the tracked object, producing a higher accuracy on the tracking.

This paper is organized as follows. In Section II, the ITWPCA algorithm is introduced. As commented before, this algorithm computes an incremental PCA matrix with the capacity of weighting every single sample. Then, in Section III, the ITWVTSP algorithm is described in detail. This VT algorithm computes the representation model of the tracked object using the ITWPCA algorithm, and assigns different importance to different regions of the tracked object. In Section IV, numerous tests are performed, showing the superiority, in terms of accuracy and stability, of the proposed approach compared to state-of-the-art techniques. Finally, in Section V, conclusion and future lines of research are derived.

Let us note that a common preliminary operation to tracking is object detection, although some algorithms perform a simultaneous detection and tracking [45], [46]. Indeed, when a tracking algorithm is intended to follow some precise objects, these objects need to be previously detected. In this paper, we do not deal with this problem and will always consider that the starting bounding box for every tracked object is given. The interested reader in object detection algorithms is referred to [47]–[51], and references inside them.

II. Incremental PCA With Weighted Samples: The ITWPCA Algorithm

In [36], an incremental PCA algorithm based on SVD update is introduced. At each iteration, the algorithm updates the existing PCA with information from the orthogonal components of the new data, with respect to the subspace generated by the PCA matrix. In [18], this algorithm was adapted to consider a changing mean of the data. Here, we add to this last algorithm the capacity of considering weights on data samples, i.e., temporal weights.

Given a set of N data samples $Z = [z_1, \ldots, z_N] \in \mathbb{R}^{M\times N}$, where each sample is represented as a vector $z \in \mathbb{R}^M$, and a weight matrix with positive elements $\Omega \in \mathbb{R}^{M\times N}$, the goal of weighted PCA is to compute the projection matrix $U \in \mathbb{R}^{M\times K}$, $K \leq N$, that minimizes the weighted squared reconstruction error

$$\tilde{\xi} = \hat{Z}^\top (I - UU^\top)^\top \Omega (I - UU^\top)\hat{Z} \tag{3}$$

where $\hat{Z}$ is the matrix obtained by subtracting the temporally weighted mean to each column of Z. The temporally weighted mean vector, μ is computed as

$$\mu = \left[\mathrm{diag}(\Omega 1_{M\times 1})\right]^{-1} \Omega Z^\top. \tag{4}$$

If only temporal weights are considered, i.e., the elements of Ω satisfy that $\omega_{ij} = \omega_{kj}\ \forall i,k \in [1,\ldots,M]$, $j \in [1,\ldots,N]$, then the weights can be expressed by a vector ${}^t\omega = [\omega_1, \ldots, \omega_N] \in \mathbb{R}^N$ and (3) can be rewritten as

$$\tilde{\xi} = \tilde{\hat{Z}}^\top (I - UU^\top)^\top (I - UU^\top)\tilde{\hat{Z}} \tag{5}$$

where $\tilde{\hat{Z}} = \sqrt{{}^t\omega}\hat{Z}$. Then, the matrix U that minimizes $\tilde{\xi}$ is composed by the K biggest eigenvectors of the covariance matrix of $\tilde{\hat{Z}}$, and can be computed by performing singular value decomposition on this matrix, i.e., $\mathrm{SVD}(\tilde{\hat{Z}}) = U\Sigma V^\top$, as introduced in [43].

For introducing the incremental version, let us first note that a scatter temporally weighted matrix S_Z, defined as

$$S_Z = \sum_{i=1}^{N} \omega_i (z_i - \mu)(z_i - \mu)^\top \tag{6}$$

differs from the weighted covariance matrix by only a scalar factor, equal to $\sum_{i=1}^{N} \omega_i$. Therefore, eigenvectors of both matrices are the same and eigenvalues are scaled by this factor, which makes equivalent to work with the covariance matrix or the scatter matrix, in terms of PCA. Let us now introduce Lemma 1.

Lemma 1: Let $Z^{(1)} = [z_1^{(1)}, \ldots, z_{N^{(1)}}^{(1)}]$ and $Z^{(2)} = [z_1^{(2)}, \ldots, z_{N^{(2)}}^{(2)}]$ be two data matrices; ${}^t\omega^{(1)} = [\omega_1^{(1)}, \ldots, \omega_{N^{(1)}}^{(1)}]$ and ${}^t\omega^{(2)} = [\omega_1^{(2)}, \ldots, \omega_{N^{(2)}}^{(2)}]$ the weights corresponding to each sample in $Z^{(1)}$ and $Z^{(2)}$, respectively; $Z^{(1,2)} = [Z^{(1)} Z^{(2)}]$ the concatenation of matrices $Z^{(1)}$ and $Z^{(2)}$; and $\mu^{(1)}$, $\mu^{(2)}$ and $\mu^{(1,2)}$ the weighted means according to ${}^t\omega^{(1)}$ and ${}^t\omega^{(2)}$ of $Z^{(1)}$, $Z^{(2)}$ and $Z^{(1,2)}$, respectively. Then, the weighted scatter matrix

Algorithm 1 Incremental Temporally Weighted PCA (ITWPCA). Given $U^{(1)}$, $\Sigma^{(1)}$, $\|{}^t\omega^{(1)}\|_1$, a forgetting factor f, and a new data matrix $Z^{(2)}$, with its corresponding weights ${}^t\omega^{(2)}$, ITWPCA computes $U^{(1,2)}$ and $\Sigma^{(1,2)}$ from the total set of data.

```
1: Compute temporally weighted mean vectors: $\mu^{(2)} = \frac{1}{\|{}^t\omega^{(2)}\|_1}\sum_{i=1}^{N^{(2)}} \omega_i^{(2)} z_i^{(2)}$ and
   $\mu^{(1,2)} = \frac{f\|{}^t\omega^{(1)}\|_1}{f\|{}^t\omega^{(1)}\|_1+\|{}^t\omega^{(2)}\|_1}\mu^{(1)} + \frac{\|{}^t\omega^{(2)}\|_1}{f\|{}^t\omega^{(1)}\|_1+\|{}^t\omega^{(2)}\|_1}\mu^{(2)}$
2: Modify new data for considering weights and the rank-1 perturbation of Lemma 1:
   $\tilde{\hat{Z}}^{(2)} = [\sqrt{\omega_1^{(2)}}(z_1^{(2)} - \mu^{(2)}), \ldots, \sqrt{\omega_{N^{(2)}}^{(2)}}(z_{N^{(2)}}^{(2)} - \mu^{(2)}), \sqrt{\frac{\|{}^t\omega^{(1)}\|_1\|{}^t\omega^{(2)}\|_1}{\|{}^t\omega^{(1)}\|_1+\|{}^t\omega^{(2)}\|_1}}(\mu^{(1)} - \mu^{(2)})]$
3: Compute orthogonal components of the modified new data: ${}^\perp\tilde{\hat{Z}}^{(2)} = orth(\tilde{\hat{Z}}^{(2)} - U^{(1)}U^{(1)\top}\tilde{\hat{Z}}^{(2)})$
4: Compute the new data matrix that will be considered for SVD: $R = \begin{bmatrix} f\Sigma^{(1)} & U^{(1)\top}\tilde{\hat{Z}}^{(2)} \\ 0 & {}^\perp\tilde{\hat{Z}}^{(2)\top}(\tilde{\hat{Z}}^{(2)} - U^{(1)}U^{(1)\top}\tilde{\hat{Z}}^{(2)}) \end{bmatrix}$
5: Compute $SVD(R) = U'\Sigma' V'$
6: Then, $U^{(1,2)} = [U^{(1)}\ {}^\perp\tilde{\hat{Z}}^{(2)}]U'$ and $\Sigma^{(1,2)} = \Sigma'$.
```

of $Z^{(1,2)}$, $S_{Z^{(1,2)}}$, can be computed as

$$\begin{aligned} S_{Z^{(1,2)}} &= S_{Z^{(1)}} + S_{Z^{(2)}} \\ &+ \frac{\|{}^t\omega^{(1)}\|_1\|{}^t\omega^{(2)}\|_1}{\|{}^t\omega^{(1)}\|_1 + \|{}^t\omega^{(2)}\|_1}(\mu^{(1)} - \mu^{(2)})(\mu^{(1)} - \mu^{(2)})^\top \end{aligned} \tag{7}$$

where $S_{Z^{(1)}}$ and $S_{Z^{(2)}}$ are the weighted scatter matrices of $Z^{(1)}$ and $Z^{(2)}$, respectively, and $\|\cdot\|_1$ denotes the 1-norm.

Its proof involves some arithmetic manipulations and can be found in [52, Sec. 3.2]. The results of Lemma 1 tell us how to express the temporally weighted scatter matrix of a big matrix by means of the weighted scatter matrices of two submatrices. Basically, the new scatter matrix is the sum of the other two, and a rank-1 perturbation that depends on the difference of means. This rank-1 perturbation can be taken into account by adding a new column to the data matrix.

Therefore, the IPCA algorithm [18] can be adapted to consider temporal weights as expressed in the incremental temporally weighted PCA (ITWPCA) algorithm described in Algorithm 1. These temporal weights modulate the contribution of every single sample to the computed PCA matrix. For instance, a sample z_i with a weight of 2 has the same impact on the PCA than having twice z_i in the data matrix.

III. Temporal and Spatial Weights in Visual Tracking: The ITWVTSP Algorithm

In VT, object representations computed *a priori*, or with some starting snapshots of the object of interest, are not robust against appearance changes along time. Ross *et al.* [18] introduced the incremental visual tracking (IVT) algorithm, a VT algorithm where the object representation, built using PCA on grayscale templates is constantly updated with the samples of the object of interest obtained by the tracker. Following the same philosophy, we introduce here the ITWVTSP algorithm. This VT algorithm exploits the capacity of the ITWPCA algorithm (introduced in Section II) for considering temporal weights for the samples added to the incremental PCA computation. The considered temporal weights are a measure of the quality of the tracked sample. This allows decreasing the impact on the PCA of bad quality tracked samples, which reduces the risk of model drift and makes the tracking more robust. Furthermore, we introduce spatial weights to favour accuracy in some predefined regions of the tracked objects, which increases tracking accuracy as will be seen in the tests. If spatial weights are not considered, which is equivalent to fixing them to one, the incremental temporally weighted visual tracking (ITWVT) algorithm is obtained. We start by introducing this algorithm in Section III-A for afterwards introducing the "Spatial Penalty" capacity in Section III-B.

Before describing in detail the ITWVTSP algorithm, let us emphasize the objectives that are followed on its development.

1) *Keep an updated model of the tracked object:* The object representation model of the tracked object is constantly updated with new samples of the object. This keeps the model constantly adapted to the current environmental and object conditions.
2) *Do not corrupt the model of the tracked object:* A quality measure of every tracked sample is computed. This measure weights the contribution of the corresponding sample to the object representation model, avoiding big impacts of bad samples on the model.
3) *Increase tracking accuracy on important regions of the tracked object:* A predefined mask can be used for giving more importance to some regions of the tracked object, such as, for instance, regions with rich texture information. This is achieved by further penalizing the tracking error on these regions, which produces a more accurate tracking.

A. Incremental Temporally Weighted Visual Tracking (ITWVT) Algorithm

A probabilistic interpretation of PCA [53] allows combining the object representation of a target, computed using PCA, with a particle filter approach for object localization. The most appropriate appearance representation for this setup is a rectangular template with affine transformations. In this case, the state space is composed of the six parameters of an affine transformation: translation (two parameters), rotation angle, scale, aspect ratio, and skew direction. The particles, which are placed in this 6-D space, represent a sample of the posterior density function of the state given the observations. Their behaviour is defined by two models: the dynamical model and the observation model. The dynamical model defines the dynamics between states, and the observation model defines the weights of particles.

Let us denote by x_k a point in the state space at time k. If no particular assumption about the allowed motion of the particles is taken, a Brownian motion can be considered. Hence, the dynamical model, $p(x_k|x_{k-1})$, is defined as

$$p(x_k|x_{k-1}) \sim N(x_k; x_{k-1}, \Theta) \tag{8}$$

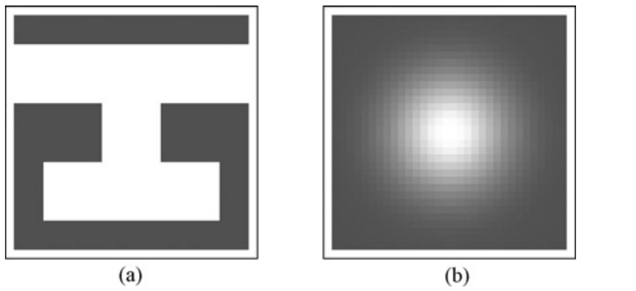


Fig. 2. Spatial weights used in the experiments. Brighter regions correspond to high weight values, darker regions to spatial weights equal to 1.0. (a) "Spec" spatial weights. (b) "Iso" spatial weights.

where Θ is a diagonal covariance matrix containing the variances of the affine parameters (translation, rotation, scale, aspect ratio, and skew direction).

In the context of PCA, the observation model gives a measure of how likely an image region, expressed as a vector $z^i \in \mathbb{R}^M$, belongs to the subspace generated by the projection matrix U, i.e., Span(U). A similar approach to condensation [16] is adopted here, assigning as weight to the particles directly their likelihood. Therefore, given an image patch z^i, a projection matrix U, a mean μ, and a diagonal matrix of eigenvalues Σ, then

$$\log p(z^i \in \text{Span}(U)) \propto -(d_t^i + d_U^i) \tag{9}$$

where d_t^i is the Euclidean distance of $z^i - \mu$ to the subspace Span(U), and d_U^i is the Mahalanobis distance within the subspace, i.e., the symmetric bilinear form defined by the inverse of the autocovariance matrix of the data. These two distances can be computed as

$$d_t^i = \frac{1}{\sigma^2}(z^i - \mu)^\top (I - UU^\top)(z^i - \mu) \tag{10}$$

where I denotes the identity matrix, and

$$d_U^i = (z^i - \mu)^\top U \Sigma^{-1} U^\top (z^i - \mu). \tag{11}$$

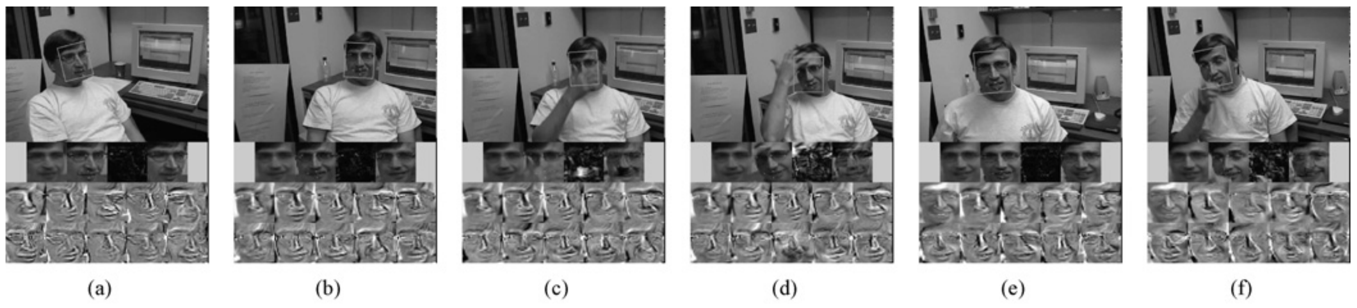
Note that since the principal components define a basis where the data is uncorrelated, the autocovariance matrix reduces to the diagonal matrix of eigenvalues Σ. The σ^2 term can be seen as the average variance lost in the projection

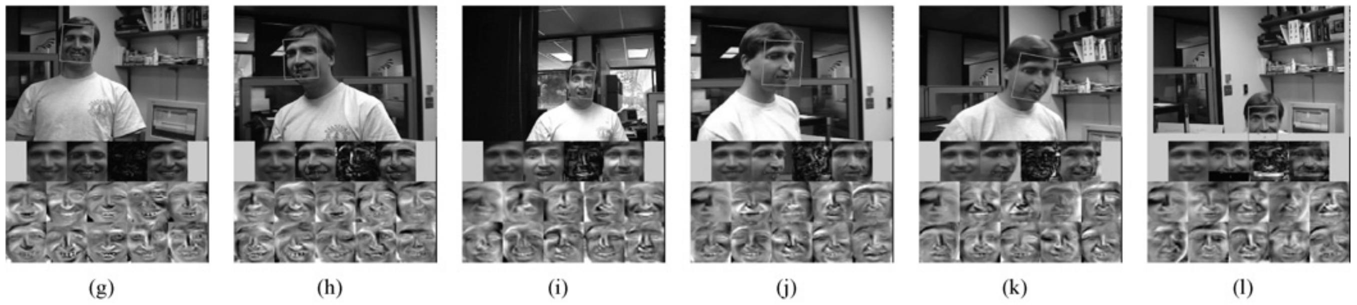
$$\sigma^2 = \frac{1}{N - N_b} \sum_{j=N_b+1}^{N} \lambda_j. \tag{12}$$

In this last equation, N_b denotes the index of the last considered eigenvector, N the total number of eigenvectors, and λ_j the eigenvalue corresponding to the jth eigenvector. Therefore, the weight of particle i at time step k before normalization is defined as

$$w_k^i = \exp\left[-(d_t^i + d_U^i)\right]. \tag{13}$$

For reducing the impact of bad samples of the tracked object, the ITWPCA algorithm is used for updating the PCA matrix. Let us introduce two measures of the quality of a tracked sample. For the sake of simplicity, and without loss of generality, images will be considered normalized, i.e., with pixel values in the interval [0, 1].





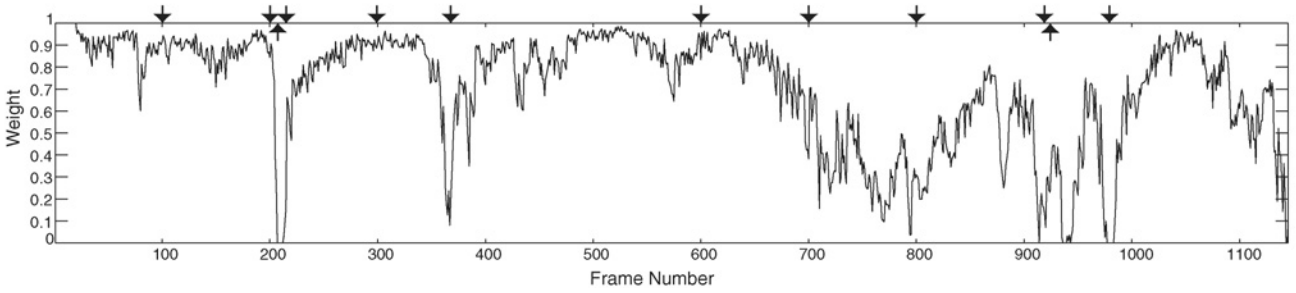


Fig. 3. Results obtained by the best run on the Dudek sequence of ITWVTSP/R-Spec with $\varepsilon = 0.07$ and ${}^{s}\omega_{\max} = 1.8$ (RMSE = 4.60px, see Table IV). The Dudek sequence presents four critical moments, approximately around frame #210, around frame #370, between frames #700 and #1000, and at the end of the sequence. In the first one, an occlusion of the face is generated with a hand (Fig. 3(c) and (d)). In the second one, the individual takes off his glasses, generating a partial occlusion of the face (Fig. 3(f)). The third critical moment is caused by a displacement of the camera and the individual, together with important out-of-the-plane rotations of the face (Fig. 3(h)–(l)). Finally, the video sequence ends with a profile view of the face. All these difficult situations are correctly handled by ITWVTSP, producing a best RMSE 26% lower than the best RMSE of IVT (29% when comparing mean RMSE values). Furthermore, the temporal weights are successfully reduced during these difficult situation (as can be seen in Fig. 3(m)), protecting the appearance model against bad samples of the tracked object. (a) Frame #100. (b) Frame #200. (c) Frame #207. (d) Frame #214. (e) Frame #300. (f) Frame #370. (g) Frame #600. (h) Frame #700. (i) Frame #800. (j) Frame #920. (k) Frame #945. (l) Frame #980. (m) Weights applied to each tracked sample of the Dudek sequence on the best run of ITWVTSP/R-Spec. The frames that present an occlusion or the frames where the face is rotated out-of-the-plane are clearly noticeable (small weights). The black arrows mark the position of the frames shown above.

Given a tracked patch expressed as a vector of pixel values $z = [z_1, \ldots, z_M]^\top \in \mathbb{R}^M$, the reconstruction error (according to the PCA matrix at this time step) gives information about the distance between the tracked patch and the subspace generated by the PCA. The difference between this patch and the PCA mean gives also information about how far the new sample is from the PCA subspace. Then, let us define the confidence on the tracked patch at time step k, c_k, as

$$c_k = \begin{cases} 1 - \frac{\alpha}{M}\sum_{i=1}^{M} f(z_i, \varepsilon), & \text{if } \sum_{i=1}^{M} f(z_i, \varepsilon) \le \frac{M}{\alpha} \\ 0, & \text{otherwise} \end{cases} \quad (14)$$

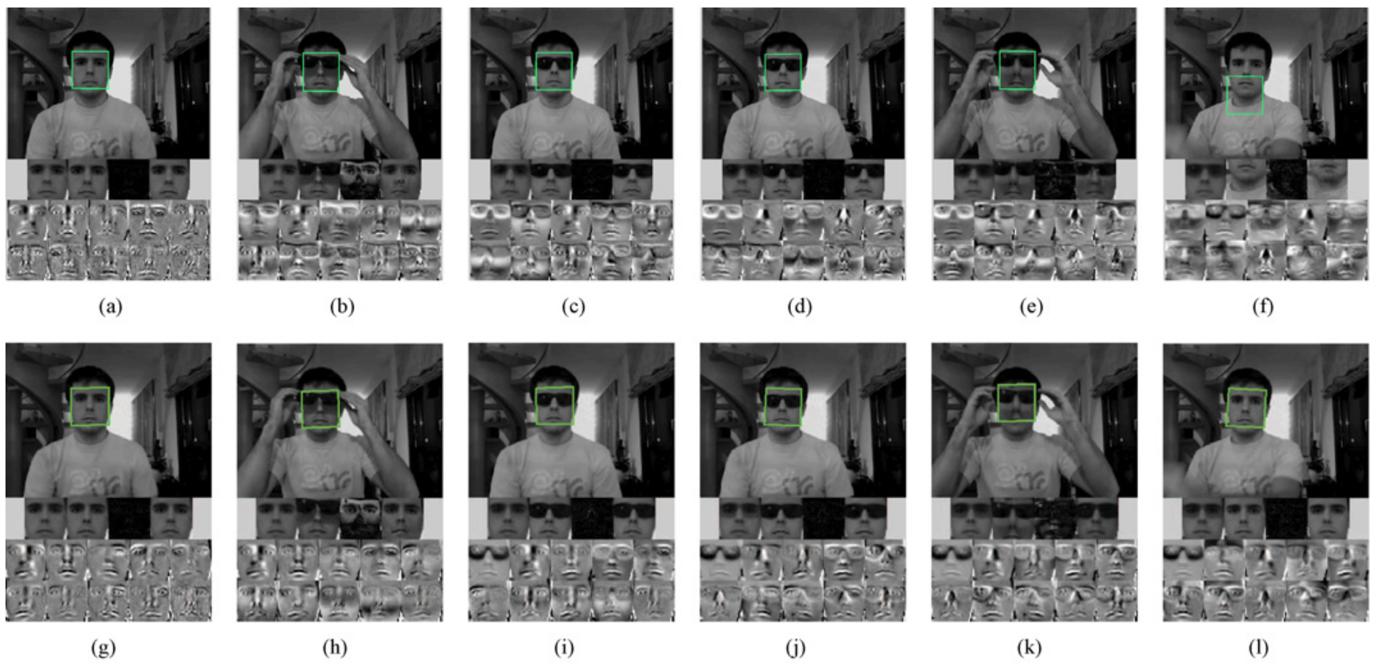
where $\varepsilon \in [0, 1]$, $\alpha \in \mathbb{R}^+$ and two different options for $f(z_i, \varepsilon)$, namely

$$f(z_i, \varepsilon) = f_R(z_i, \varepsilon) = \begin{cases} 1, & \text{if } |(z_i - \mu_i) - \bar{z}_i| \ge \varepsilon \\ 0, & \text{otherwise} \end{cases} \quad (15)$$

and

$$f(z_i, \varepsilon) = f_M(z_i, \varepsilon) = \begin{cases} 1, & \text{if } |z_i - \mu_i| \ge \varepsilon \\ 0, & \text{otherwise} \end{cases} \quad (16)$$

being $\bar{z}_i$ the ith component of the vector $\bar{z} = UU^\top(z - \mu)$. The measure proposed in (14) gives more importance to the number of pixels with a significant error (ε) than to the amount of the error itself. Furthermore, samples with more than $\frac{100}{\alpha}\%$ of the pixels with an error higher than ε are discarded ($\omega = 0$). This strategy tries to penalize samples, containing big regions with a significant amount of error (reconstruction error or distance to the mean). Indeed, this is the typical situation when, for instance, a region of the tracked object is occluded by another object. The neutral value of $\alpha = 2$ has been adopted in all the tests, i.e., samples with more than 50% of the pixels with a significant error are not considered in the PCA computation. Depending on the context of the application, this value can be decreased. However, a value too low (α too high) would



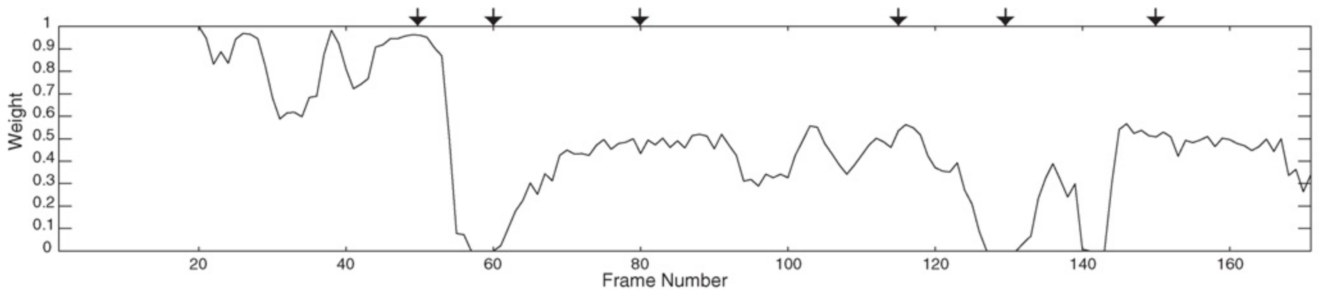


Fig. 4. Results obtained with the IVT and the ITWVT/M algorithms on the *Rockstar* sequence. The eigenvectors show how the sunglasses corrupt the appearance model computed using IVT, avoiding a correct tracking continuation after taking them off. In the case of ITWVT/M, the information of the sunglasses has a much lower impact on the appearance model, as can be observed in the eigenvectors. This is achieved thanks to the low weight values for the corresponding object samples. (a) IVT: Frame #50. (b) IVT: Frame #60. (c) IVT: Frame #80. (d) IVT: Frame #115. (e) IVT: Frame #130. (f) IVT: Frame #150. (g) ITWVT/M: Frame #50. (h) ITWVT/M: Frame #60. (i) ITWVT/M: Frame #80. (j) ITWVT/M: Frame #115. (k) ITWVT/M: Frame #130. (l) ITWVT/M: Frame #150. (m) Weights applied by the ITWVT/M algorithm to every tracked sample of the *Rockstar* sequence. The black arrows mark the position of the frames shown above. The big changes of appearance caused by putting on and taking off the sunglasses are clearly detected by ITWVT/M, approximately around frames #60 and #130, respectively. Indeed, weights around these frames are decreased to 0. After these changes, it can be observed how weights are progressively increased, which allows a slow introduction of new information to the appearance model, without corrupting it.

generate a risk of being too restrictive when accepting new samples on the PCA computation. This could easily get the model unadapted to the object of interest.

B. Adding Spatial Weights: The ITWVTSP Algorithm

In a VT application, all the sensors that feed variables (pixel sensors) are supposed to be identical, which makes spatial weights in the PCA estimation process not as easy to interpret as temporal weights. Indeed, this weighting would give different importance to pixels depending on the pixel sensor, while all sensors are supposed to be identical. In fact, in VT the important thing is the accuracy in the tracking of certain regions of the object of interest, not the accuracy of the model for these regions. For instance, if a face is being tracked, special care must be taken to correctly track the regions containing more information (eyes, nose, and mouth), but a correct delimitation of the cheek is not as important, in general. Furthermore, contrary to temporal weights, spatial weights cannot be added to the PCA computation by performing a pre-processing of the input data. A way of computing a PCA matrix taking into account spatial weights is by performing an iterative optimization process on (3). This is the approach adopted for instance in [44]. However, this is not well adapted to a VT context, where performances close to real time and bounded computation time per frame are required.

Let us propose here a way of achieving an increase of the tracking accuracy of important regions of tracked objects, without requiring a modification of the appearance model. This can be accomplished by penalising the contribution of pixels in important regions to the distances in (10) and (11), i.e., by applying a spatial penalty to hypothesis. Let us define a vector of positive values ${}^{s}\omega \in \mathbb{R}^M$ as the desired spatial weights, i.e., pixel weights. The higher the value applied to a pixel, the higher the penalty applied to this pixel and therefore more importance assigned to this pixel. Indeed, hypothesis fitting better these more penalized pixels will be favored. Thus, let

Algorithm 2 Incremental Temporally Weighted Visual Tracking with Spatial Penalty (ITWVTSP). The target region (image of the object in the first frame) is denoted by z_0; $N^{(2)}$ denotes the size of the processed blocks; f denotes the forgetting factor; and K denotes the maximum number of considered eigenvalues.

```
1:  $\mu = z_0$, $n = 1$, and $U^{(1)}$, $\Sigma^{(1)}$, $Z^{(2)}$ and ${}^t\omega^{(2)}$ are empty
2:  Set ${}^s\omega$ to the desired spatial weights (by default, ${}^s\omega = 1_{M\times 1}$)
3:  for every frame of the video do
4:    Draw particles according to the dynamical model (8) and the weight distribution of particles.
5:    For each particle, compute its weight according to the observation model and spatial weights ((17) and (18)).
6:    Store in $Z^{(2)}$ the image region corresponding to the most likely particle, and in ${}^t\omega^{(2)}$ its PCA weight ((14))
7:    if there are $N^{(2)}$ stored images in $Z^{(2)}$ then
8:      if $n < K$, i.e., if the effective number of samples is smaller than $K$ then
9:        All the samples are considered and with a temporal weight equal one: ${}^t\omega_i^{(2)} = 1, \ \forall i = 1, \ldots, N^{(2)}$
10:     end if
11:     Apply Algorithm 1 with $\|{}^t\omega^{(1)}\|_1 = n$, discarding the eigenvectors that exceed $K$.
12:     Set $U^{(1)} = U^{(1,2)}$, $\Sigma^{(1)} = \Sigma^{(1,2)}$ and $n = fn + \|{}^t\omega^{(2)}\|_1$
13:     Empty the matrix $Z^{(2)}$ and the vector ${}^t\omega^{(2)}$
14:   end if
15: end for
```

us redefine (10) and (11) by considering spatial weights as

$$d_t = \frac{1}{\sigma^2}(z-\mu)^\top {}^s\Omega(I - UU^\top){}^s\Omega(z-\mu) \qquad (17)$$

$$d_U = (z-\mu)^\top {}^s\Omega U \Sigma^{-1} U^\top {}^s\Omega(z-\mu) \qquad (18)$$

where ${}^s\Omega = diag({}^s\omega)$ is a diagonal matrix with the spatial weights, and σ^2 is defined in (12). The expression in (17) computes a weighted Euclidean distance to the subspace generated by the PCA, while (18) computes a weighted Mahalanobis distance within this subspace.

The use of (17) and (18) for computing particle weights takes into account the importance given beforehand to every single pixel of the tracked region. This implies that the values of individual pixels of every hypothesis have different importance in the computation of the particle weight. However, an important thing to have in mind is that an excessive increase of the weight applied to certain pixels can render the tracking algorithm unstable (as will be shown in Section IV).

In Algorithm 2, a detailed description of the complete proposed visual tracking algorithm with incremental temporally weighted PCA and spatial error penalty is shown. Note that by fixing ${}^s\omega = 1_{M\times 1}$, we obtain the ITWVT algorithm. We denote by "/R" the use of (15) and by "/M" the use of (16), i.e., for instance we denote by ITWVT/M the ITWVTSP algorithm using $f_M(z, \varepsilon)$ and ${}^s\omega = 1_{M\times 1}$.

IV. Tests and Results

We have performed several tests, on several video sequences, to the ITWVT and the ITWVTSP algorithms. For showing the improvement obtained by the weighting strategy, the results are compared with the results obtained by the IVT algorithm introduced in [18].[1] For a general comparison against state-of-the-art algorithms, we compare our results with the results obtained using the TLD algorithm introduced in [15].[2]

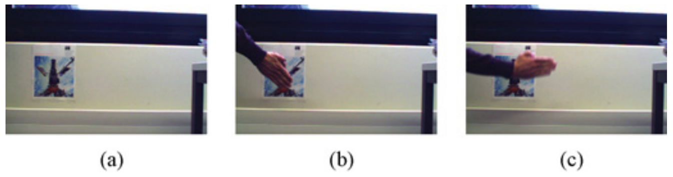


Fig. 5. Several frames of the *Poster* sequence. The total sequence is composed of 585 frames. (a) Frame #100. (b) Frame #161. (c) Frame #310.

All the common parameters between IVT, ITWVT, and ITWVTSP are fixed to the same values, taken from those proposed in [18], i.e., 600 particles, an eigenvector size of 32×32 pixels, a maximum number of 16 eigenvectors and a block update of five images. We only increase slightly the forgetting factor (from 0.95 to 0.97) since the temporal weights increase the quality of the model and a longer memory is beneficial. The standard deviations of the dynamical model (8) in all the experiments are $9.0px$ for row and column displacements, 0.05 radians for rotation, 0.05 for scaling in the x direction, 0.001 for scaling in the y direction, and 0.001 radians for the scaling angle defining x and y directions. These are similar values to those proposed in the implementation of IVT. Since IVT can be considered a particular case of ITWVT (fixing ε to 1 in (14)), and ITWVT a particular case of ITWVTSP (fixing ${}^s\omega$ to $1_{M\times 1}$), the use of the same common parameters shows the improvement provided by the temporal weighting in ITWVT and the spatial weighting in ITWVTSP. Note that with these parameters, the implementation of ITWVTSP in MATLAB runs at 7 frames per second in a laptop with a 2.0-GHz processor.

With respect to TLD, the standard parameters provided in the distributed implementation are used.

For visualization of the tracking results, we use the same template as in [18]: the first row contains the current frame with the tracked region, the second row contains the mean, the tracked window, the reconstruction error and the reconstructed image, and finally, the third and fourth rows contain the first ten eigenvalues (Figs. 3, 4, 8–10).

The performed experiments are divided into two groups. In the first group, there are experiments performed on labeled video sequences, i.e., video sequences with a ground truth. In these experiments, quantitative performance scores are computed to show the performance of the algorithms. In the second group, the proposed algorithms are applied to several unlabeled video sequences in a variety of tracking applications, to show the polyvalence of ITWVT and ITWVTSP.

Note that given the implicit stochasticity of all the algorithms (IVT, TLD, ITWVT, and ITWVTSP) each quantitative

[1]Implementation available at http://www.cs.toronto.edu/~dross/ivt/ (last visited in May 2012).

[2]Implementation available at http://info.ee.surrey.ac.uk/Personal/Z.Kalal/tld.html (last visited in May 2012).

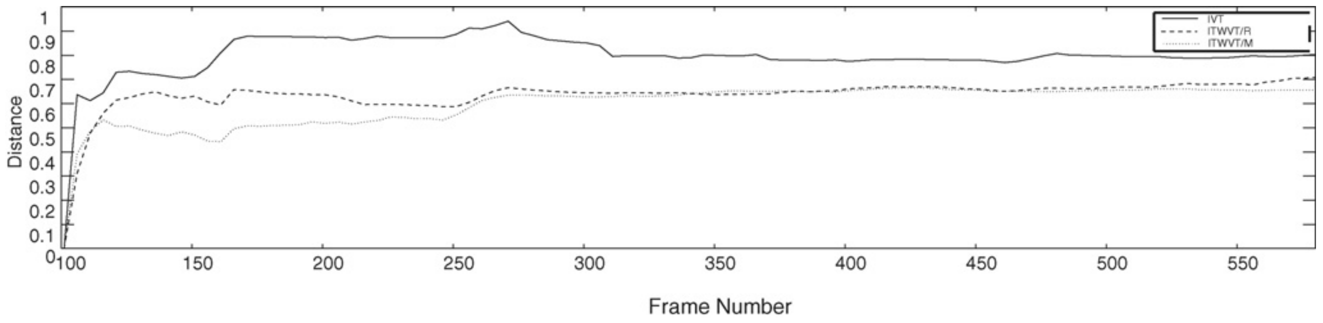


Fig. 6. Distances between the first eigenvector at frame 100 and the first eigenvector computed using IVT (solid line), ITWVT/R (dashed line) and ITWVT/M (dotted line) at subsequent frames. It can be observed that the deviation from the model before introducing occlusions is lower when ITWVT/R or ITWVT/M are used.

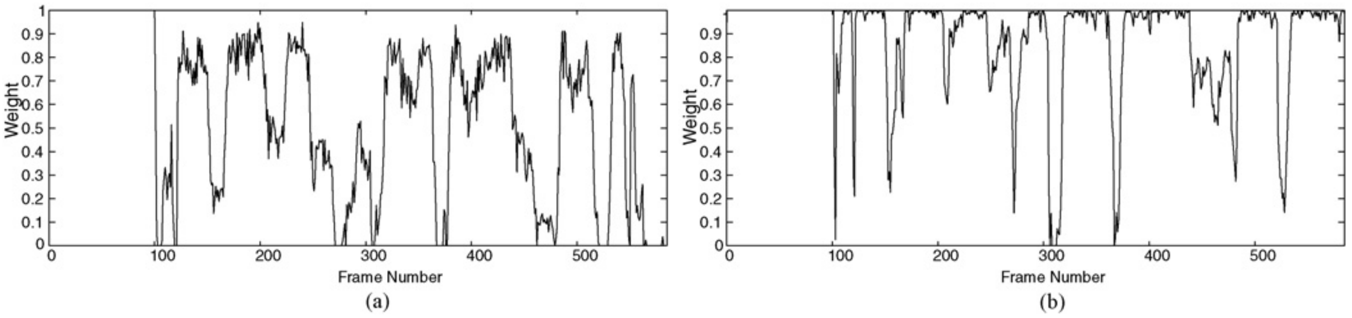


Fig. 7. Weights applied to the samples of the poster in the *Poster* sequence using ITWVT/M and ITWVT/R. The abrupt reductions of temporal weight values correspond to frames where the poster is occluded (Fig. 5(b) and (c) for two examples). (a) ITWVT/M. (b) ITWVT/R.

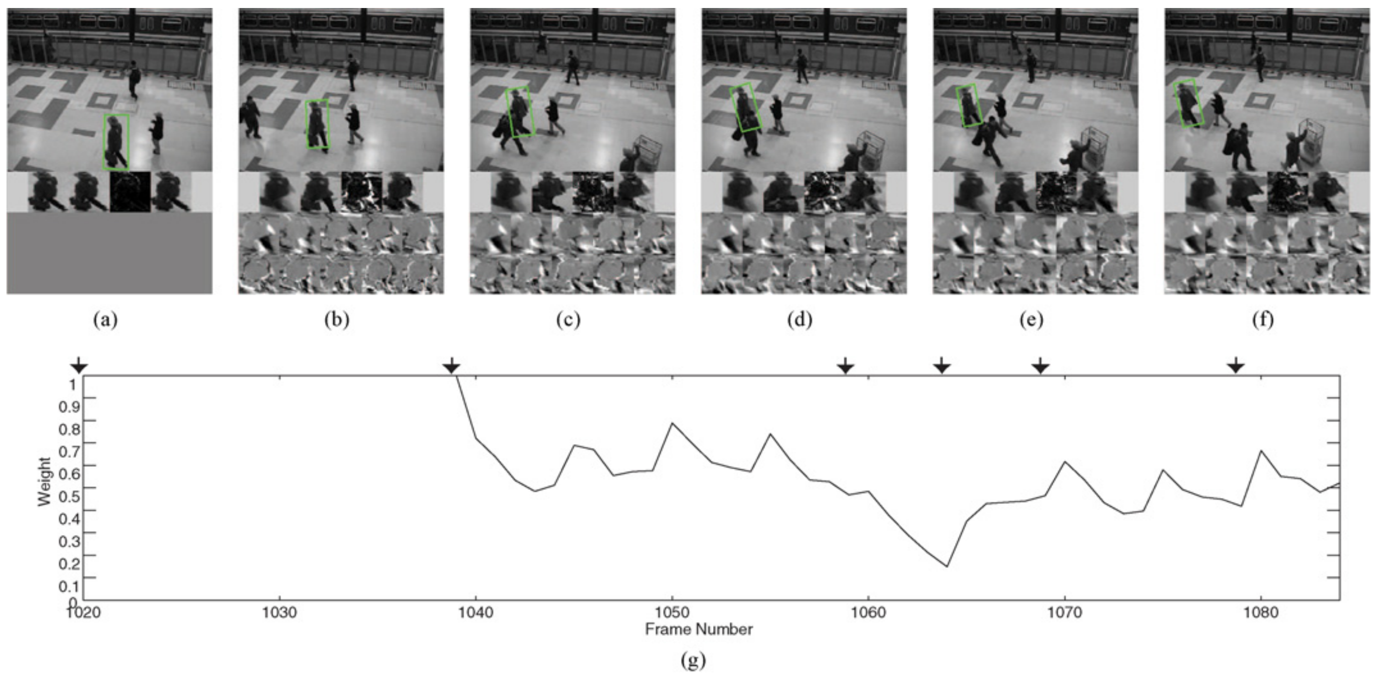


Fig. 8. Example of *Pedestrian* tracking using ITWVTSP/R-iso (ε = 0.12 and ${}^{s}\omega_{\max}$ = 3.2) on the sequence S1-T1-C Camera 3 of the PETS2006 Dataset. (a) Frame #1020. (b) Frame #1039. (c) Frame #1059. (d) Frame #1064. (e) Frame #1069. (f) Frame #1079. (g) Weights applied to each frame. The black arrows mark the position of the frames shown above. The partial occlusion generated by a pedestrian walking in the opposite direction (around frame #1064) can be observed in the weights, which are reduced up to less than 0.2.

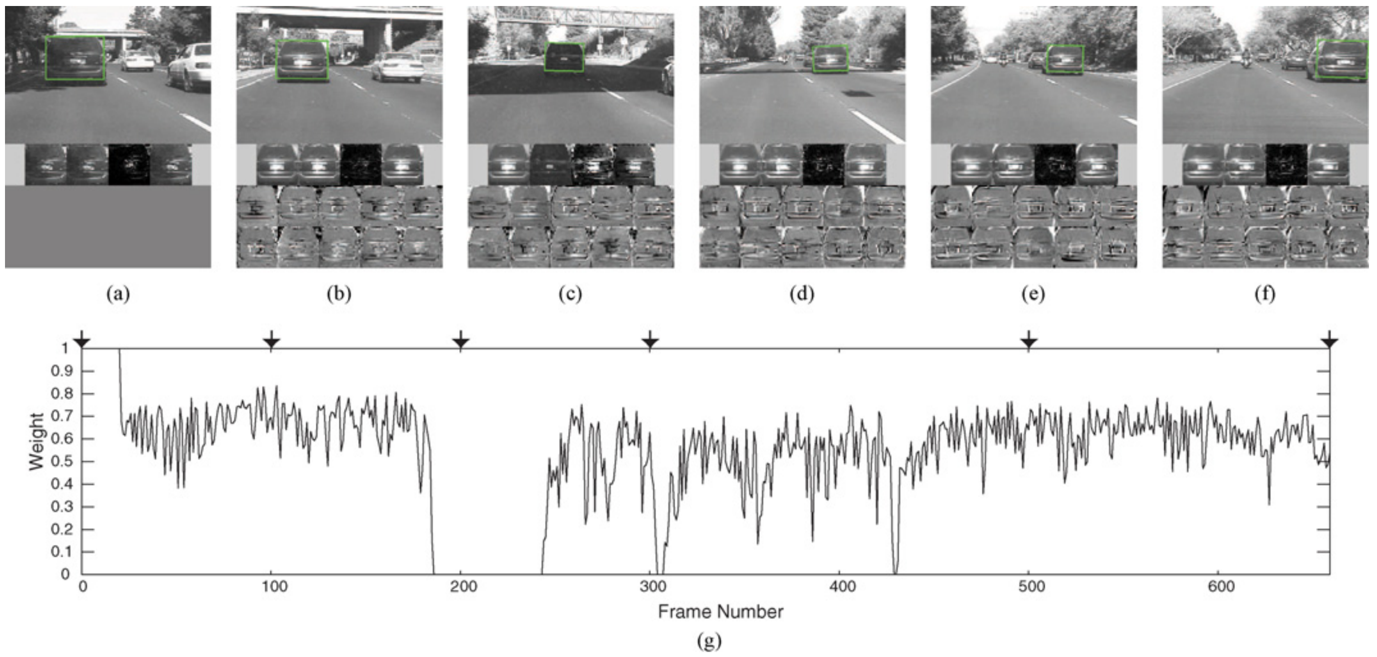


Fig. 9. Example of vehicle tracking using ITWVTSP/R-iso ($\varepsilon = 0.07$ and ${}^{s}\omega_{\max} = 2.0$). The vehicle passes under a bridge between frames #190 and #240, approximately (Fig. 9(b)). This causes a sudden change on the appearance of the tracked vehicle during the above-mentioned frames. ITWVTSP correctly detects this appearance change and reduces the temporal weights of the corresponding samples (Fig. 9(g)), protecting the appearance model of bad samples while pursuing the tracking without problems. Two more sudden changes of appearance are present in the video sequence, around frame #305 (Fig. 9(d)) and around frame #430. In this case, the changes are caused by strong shadows and are also correctly handled by ITWVTSP. (a) Frame #1. (b) Frame #100. (c) Frame #200. (d) Frame #300. (e) Frame #500. (f) Frame #650. (g) Weights applied to each frame. The black arrows mark the position of the frames shown above.

score is computed considering the results of ten independent runs of the corresponding algorithm.

A. Labeled Video Sequences

The *Dudek* sequence [54] is a very challenging video sequence where the tracking of a face can be performed. The sequence consists of 1145 frames with changes in the tracked object, the camera position, and the illumination (see Fig. 3). It is publicly available, together with the ground truth of seven manually labeled points on the face, which makes it widely used for quantitative comparisons between tracking algorithms. Indeed, the root mean squared error (RMSE) in pixels of the tracked points with respect to the real ones (ground truth) can be computed to quantify the tracking accuracy. In our tests, the runs that produce a mean RMSE among all the frames bigger than 10.0px are considered losses of track, and therefore not taken into account in the RMSE statistics.

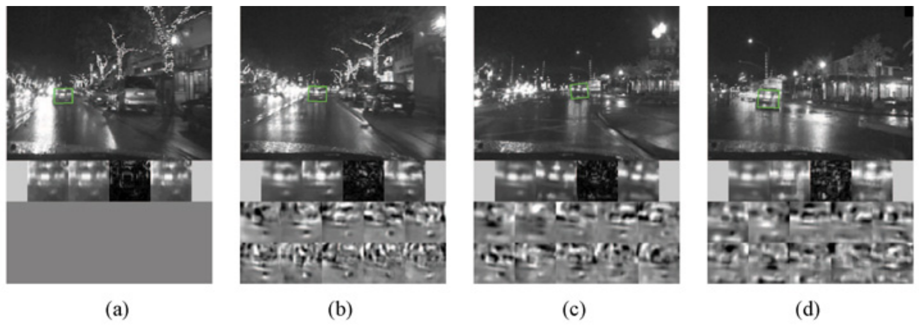
Ten runs of IVT have been performed on this sequence. The best run produces a RMSE of 6.23px, and among the ten runs, five are considered losses of track (RMSE > 10.0px). The complete statistics are shown in the first row of Table IV.

The ITWVT algorithm has two variants, depending on the use of (15) or (16). Both variants are controlled by one additional parameter with respect to IVT, the ε parameter in the above-mentioned equations. For fixing this parameter, 25 different values of $\varepsilon \in [0.01, 0.9]$ have been tested, launching ten independent runs on the *Dudek* sequence for each one of them. ITWVT/M and ITWVT/R produce very similar results, with small variance among runs with $\varepsilon \in [0.02, 0.12]$ (Table I). This is due to the fact that small values of ε produce low temporal weights, avoiding a good adaptation of the model to the tracked face, while big values of ε produce big weights, making the performance similar to IVT (in terms of RMSE and number of track losses). For $\varepsilon \in [0.02, 0.12]$, the compromise between good model adaptation and corruption avoidance seems to be satisfied for the *Dudek* sequence.

A reasonable value for the error threshold is $\varepsilon = 0.07$. For this value, ITWVT/M obtains a best run with RMSE = 5.65px and ITWVT/R a best run with RMSE = 5.86px, both with only one out of the ten runs losing the track. This shows how the addition of the temporal weight (with respect to IVT) improves the tracking (lower RMSE) at the same time that makes it more robust (less losses of track). Although the performances of ITWVT/R and ITWVT/M are similar, comparing the obtained temporal weights it can be observed that weights obtained using the reconstruction error are more consistent. Indeed, only frames with an occlusion or high out-of-plane rotations of the face present a clearly reduced weight in ITWVT/R.

Let us note that the value of ε is application specific. Indeed, the appearance of a rigid object changes slightly, which allows fixing a more restrictive (smaller) ε. On the contrary, a deformable object, such as a pedestrian, changes its appearance considerably, which forces to fix ε to higher values if we want to avoid unjustified small temporal sample weights. Faces are somehow in between highly deformable objects and rigid objects, which makes $\varepsilon = 0.07$ an appropriate candidate value when no information about the application is available.

Once ε is fixed to 0.07, the ${}^{s}\omega$ vector has to be defined to use ITWVTSP. Two different vectors have been designed for



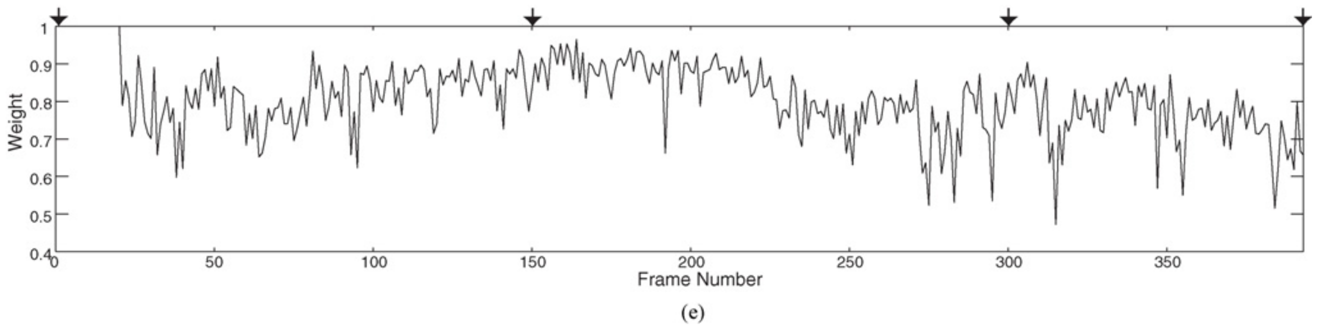


Fig. 10. Example of night vehicle tracking using ITWVTSP/R-iso (ε = 0.12 and ${}^{s}\omega_{\max}$ = 2.0). In this sequence, the illumination of the scene is constantly bad, without any worsening nor improvement (Fig. 10(a)–(d)). ITWVTSP correctly tracks the vehicle during the whole sequence, applying similar temporal weights to all the samples, as expected (Fig. 10(e)). (a) Frame #1. (b) Frame #150. (c) Frame #300. (d) Frame #390. (e) Weights applied to each frame. The black arrows mark the position of the frames shown above.

the tests. The first one assigns a high weight value to pixels on important regions of the face and 1.0 to the rest [Fig. 2(a)]. We call these weights the "spec" spatial weights and denote its use by "-spec". The second one is a two-dimensional Gaussian shape centred in the middle of the patch [Fig. 2(b)]. We call these weights the "iso" spatial weights and denote its use by "-iso." The "iso" spatial weights consider that pixels far from the boundary of the tracked object are more important. Both weight vectors, "spec" and "iso," can be defined by means of a single parameter (${}^{s}\omega_{\max}$), which is the value of the maximum spatial weight (the minimum is always 1.0).

Following the same procedure than before, for each variant of the algorithm (ITWVTSP/M-spec, ITWVTSP/M-iso, ITWVTSP/R-spec and ITWVTSP/R-iso), ten independent runs have been performed on the Dudek sequence for 12 different values of ${}^{s}\omega_{\max} \in [1.2, 3.4]$. For values of ${}^{s}\omega_{\max} > 2.0$ using "spec," the algorithm starts to be unstable, producing more losses of track than correct tracking among the ten performed runs. For the "iso" spatial weights, the gradual transition of the weights makes the algorithm more stable, allowing to go up to ${}^{s}\omega_{\max}$ = 3.2. Good values for the maximum spatial weights are 1.8 for "spec" and 3.2 for "iso," although smaller values can be used if we want to minimise the risk of loss of track due to excessive spatial penalty. The complete statistics of the obtained results are shown in Table II for ITWVTSP/M, and in Table III for ITWVTSP/R. The results obtained by the best run of ITWVTSP/R-Spec with ε = 0.07 and ${}^{s}\omega_{\max}$ = 1.8 are shown in Fig. 3.

A comparison of the results obtained by IVT; ITWVT with ε = 0.07; ITWVTSP/M-iso and ITWVTSP/R-iso with ε = 0.07 and ${}^{s}\omega_{\max}$ = 3.2; and ITWVTSP/M-spec and ITWVTSP/R-spec with ε = 0.07 and ${}^{s}\omega_{\max}$ = 1.8 is shown in Table IV. As it can be observed, the ITWVTSP algorithm produces a considerable better tracking performance than IVT with, at the same time, an increased robustness (two out of ten track losses for ITWVTSP against five out of ten for IVT). The statistics using TLD are not shown in this table because all the runs produce a RMSE higher than 10.0px. Indeed, TLD tends to enlarge or reduce the tracked region on the Dudek sequence, which causes a displacement on the template of the tracked points and therefore a high RMSE value.

For testing the algorithms in a real situation with partial occlusions, we recorded the *Rockstar* sequence. This sequence is composed of 171 frames and shows a human being in front of the camera. At a certain moment, the recorded person puts on a pair of sunglasses that he takes off later. These sunglasses generate an occlusion of the eyes of the subject, which is an important part of the face clearly coded in the appearance model. The distance between the face of the subject and the camera, and therefore its size in the image, remains almost constant during the whole video. This allows to label the ground truth of the sequence by displacing the starting bounding box that contains the face, to keep eyes, nose and mouth centred along the whole video sequence. This has been done manually for generating the ground truth.

Ten runs of TLD, IVT, ITWVT/{M,R} and ITWVTSP/{M,R}-{iso,spec} have been performed on this sequence. For temporal weights, ε has been fixed to 0.07, and for spatial weights a conservative approach has been adopted, taking ${}^{s}\omega_{\max}$ = 2.0 for "iso" and ${}^{s}\omega_{\max}$ = 1.6 for "spec". Precision and lost track ratio scores [10] have been computed for all the algorithms. The results are shown in Table V. For computing

TABLE I

STATISTICS OF THE OBTAINED RESULTS ON THE DUDEK SEQUENCE USING ITWVT/M AND ITWVT/R VARYING ε

ε Value	ITWVT/M					ITWVT/R				
	Losses of Track	RMSE				Losses of Track	RMSE			
		Mean	Worst	Best	StdDev		Mean	Worst	Best	StdDev
0.01	3	8.4058	9.7601	7.6145	0.7917	2	6.4237	9.0563	5.4594	1.3503
0.02	2	6.3769	7.7979	5.8036	0.6203	0	7.1984	7.8388	6.3539	0.5411
0.03	3	6.2378	7.3919	5.6531	0.6865	0	6.7750	7.3613	6.3504	0.3254
0.04	0	7.3831	8.4052	6.5772	0.5698	0	6.3040	7.0071	5.9941	0.3099
0.05	0	7.2318	8.3727	6.4899	0.5656	0	6.5492	7.3757	6.0981	0.4334
0.06	0	7.0817	8.3856	6.4095	0.5787	0	6.5031	7.2581	5.8174	0.3876
0.07	1	6.5527	7.5133	5.6537	0.6473	1	6.6765	7.8109	5.8645	0.7013
0.08	0	6.5587	7.2342	6.0085	0.3791	1	6.7496	7.4763	6.1413	0.4116
0.09	1	6.3484	6.9601	5.8681	0.3763	2	7.3782	9.6218	6.4713	1.0165
0.10	2	6.5412	7.0748	5.6764	0.5273	4	6.5855	7.0886	6.2977	0.2958
0.12	1	6.6076	7.0852	6.2457	0.3150	1	6.9439	8.0860	6.1087	0.6061
0.14	1	6.8099	7.5038	6.2408	0.4458	3	7.0105	8.7338	6.1794	0.8975
0.16	3	6.4705	6.8153	6.1707	0.2097	2	7.2846	8.8147	6.1709	0.9767
0.18	4	7.2468	8.5889	6.1936	0.8449	4	6.6198	8.1372	6.1489	0.7598
0.20	3	6.5875	7.2460	5.6807	0.5313	4	6.9906	7.8886	6.1627	0.7460
0.25	4	6.7973	7.3008	6.1397	0.4934	2	7.1186	8.9511	6.0767	0.8914
0.30	3	7.0953	9.9336	5.9823	1.3220	2	7.0153	8.7097	6.0319	0.8136
0.35	0	7.0374	8.4965	6.3425	0.7452	4	6.9057	7.4181	6.6161	0.2888
0.40	5	6.8801	7.5075	6.3805	0.5100	5	6.8099	7.5384	6.1242	0.5307
0.45	1	6.8813	8.0999	6.2154	0.5896	3	6.7475	7.5293	5.7280	0.5996
0.50	5	6.4757	7.0297	6.0850	0.3577	6	6.8891	7.9685	6.0177	0.8847
0.60	1	6.9352	8.0884	6.2065	0.6743	1	6.7079	7.4916	5.7971	0.5352
0.70	3	6.5571	7.5610	6.0677	0.5217	3	6.6355	7.2467	6.2273	0.3910
0.80	4	6.6718	7.2075	6.3131	0.3549	5	6.7616	7.5518	6.0889	0.6239
0.90	5	7.9066	9.9127	6.2822	1.4074	3	7.0418	7.9330	6.3948	0.6415

TABLE II

STATISTICS OF THE OBTAINED RESULTS ON THE DUDEK SEQUENCE USING ITWVTSP/M-SPEC AND ITWVTSP/M-ISO WITH $\varepsilon = 0.07$ AND VARYING ${}^{s}\omega_{\max}$

Max Spatial Weight	ITWVTSP/M-Spec					ITWVTSP/M-Iso				
	Losses of Track	RMSE				Losses of Track	RMSE			
		Mean	Worst	Best	StdDev		Mean	Worst	Best	StdDev
1.2	3	6.1157	6.5954	5.6980	0.3029	1	6.3395	7.3405	5.6972	0.4703
1.4	2	5.8037	7.2448	5.1832	0.6446	1	6.0340	6.7508	5.7090	0.3229
1.6	1	5.2282	6.0723	4.9648	0.3404	0	6.0685	6.5282	5.6664	0.2504
1.8	2	5.2210	5.9466	4.7596	0.3755	0	5.8406	6.7670	5.2946	0.4247
2.0	5	4.9565	5.1429	4.7190	0.1765	0	5.6698	6.3455	5.3115	0.3508
2.2	6	5.2040	5.8718	4.8028	0.5013	0	5.7224	6.1170	5.2639	0.2684
2.4	6	4.9988	5.5915	4.7908	0.3953	2	5.7545	6.3612	5.2255	0.3813
2.6	4	5.6069	8.7299	4.7570	1.5463	1	5.5202	6.2085	5.1877	0.3344
2.8	6	5.3201	5.8650	4.7597	0.4560	1	5.6210	6.7804	5.0467	0.5699
3.0	7	5.7216	7.3102	4.7601	1.3859	1	5.6613	6.3716	5.3092	0.3519
3.2	8	5.0908	5.1136	5.0680	0.0322	2	5.4659	6.2991	4.9586	0.4604
3.4	9	8.5606	8.5606	8.5606	0.0000	8	5.3848	5.7800	4.9896	0.5589

TABLE III

STATISTICS OF THE OBTAINED RESULTS ON THE *Dudek* SEQUENCE USING ITWVTSP/R-SPEC AND ITWVTSP/R-ISO WITH $\varepsilon = 0.07$ AND VARYING ${}^{s}\omega_{\max}$

Max Spatial Weight	ITWVTSP/R-Spec					ITWVTSP/R-Iso				
	Losses of Track	RMSE				Losses of Track	RMSE			
		Mean	Worst	Best	StdDev		Mean	Worst	Best	StdDev
1.2	0	5.8092	6.5682	5.1404	0.4489	1	6.5534	7.2461	6.0495	0.4174
1.4	0	5.7330	6.7432	5.1634	0.5469	3	6.7407	9.3743	5.6009	1.2095
1.6	0	5.3833	6.5522	4.6985	0.5362	2	6.2685	7.2531	5.4140	0.6783
1.8	2	4.8869	5.4463	4.5969	0.2637	1	6.2240	9.6946	5.0292	1.3994
2.0	2	5.1666	5.9561	4.7369	0.3956	0	5.5857	5.9252	5.0960	0.2718
2.2	5	5.3154	5.6749	4.8982	0.2972	0	5.3548	5.9136	4.6412	0.4747
2.4	6	6.4246	7.3709	5.5164	0.9404	0	5.2664	6.4271	4.7385	0.4663
2.6	5	5.9692	7.2687	5.5126	0.7376	0	5.2294	5.5833	4.8788	0.2327
2.8	4	6.0234	8.3291	5.3102	1.1399	0	5.2520	5.9808	4.9387	0.3533
3.0	2	7.0825	9.8382	5.5355	1.5283	0	5.0500	5.3742	4.6198	0.2461
3.2	2	6.1169	7.2153	5.1857	0.8506	2	5.2135	5.6375	4.6927	0.3145
3.4	5	6.5191	7.3183	5.2347	0.8819	8	5.4944	5.8743	5.1145	0.5372

TABLE IV
STATISTICS OF THE OBTAINED RESULTS ON THE *Dudek* SEQUENCE. THE PARAMETERS ARE $\varepsilon = 0.07$, ${}^{s}\omega_{\max} = 3.2$ FOR "ISO" AND ${}^{s}\omega_{\max} = 1.8$ FOR "SPEC"

Algorithm	Losses of Track	Mean RMSE	Worst RMSE	Best RMSE	StdDev RMSE
IVT	5	6.8702	7.2790	6.2324	0.3964
ITWVT/R	**1**	6.6765	7.8109	5.8645	0.7013
ITWVT/M	**1**	6.5527	7.5133	5.6537	0.6473
ITWVTSP/M-iso	2	5.4659	6.2991	4.9586	0.4604
ITWVTSP/M-spec	2	5.2210	5.9466	4.7596	0.3755
ITWVTSP/R-iso	2	5.2135	5.6375	4.6927	0.3145
ITWVTSP/R-spec	2	**4.8869**	**5.4463**	**4.5969**	**0.2637**

TABLE V
OBTAINED RESULTS ON THE *Rockstar* SEQUENCE

Algorithm	Precision				Lost Track Ratio			
	Best	Worst	Mean	St.Dev.	Best	Worst	Mean	St.Dev.
IVT	0.9064	0.3392	0.6526	0.1877	0.0	0.1637	0.0965	0.0831
TLD	0.6260	0.3493	0.5598	0.0833	0.1053	0.2398	0.1632	0.0671
ITWVT/R	0.9591	0.3392	0.8474	0.1859	**0.0**	**0.0117**	**0.0012**	**0.0037**
ITWVT/M	**0.9708**	**0.7661**	**0.9129**	**0.0739**	0.0	0.0760	0.0111	0.0241
ITWVTSP/M-iso	0.7895	0.3509	0.6018	0.1239	0.0	0.1579	0.0287	0.0518
ITWVTSP/M-spec	0.9532	0.7602	0.8111	0.0753	0.0	0.0877	0.0322	0.0391
ITWVTSP/R-iso	0.8187	0.3333	0.5468	0.1752	0.0	0.1579	0.0503	0.0744
ITWVTSP/R-spec	0.9825	0.6842	0.7754	0.0813	0.0	0.1637	0.0830	0.0733

The parameters are $\varepsilon = 0.07$, ${}^{s}\omega_{\max} = 2.0$ for "iso" and ${}^{s}\omega_{\max} = 1.6$ for "Spec."

precision score, the intersection over union criterion, with a threshold value of 0.8, has been used. For the lost track ratio, dice error with a threshold value of 0.8 has been employed. The results show clearly the better performance of the family of algorithms introduced (ITWVT and ITWVTSP). However, the negative impact in this case of the spatial penalty can be observed too. Indeed, the persistence of the partial occlusion in an important region (the eyes) seems to have a negative effect in the performance due to spatial weights, although it is anyway better than with IVT and TLD. These two algorithms suffer from a displacement of the tracked region while the subject is wearing the sunglasses, which causes the bad precision and lost track ratio scores. Some selected frames of the best run using IVT and ITWVT/M are shown in Fig. 4.

B. *Unlabeled Video Sequences*

In Fig. 5, several frames of the poster sequence are shown. In this sequence, a poster is recorded while several partial occlusions are generated. The total sequence is composed of 585 frames, and during the first 100 frames there are no occlusions. In order to see the effect of temporal weights, we compute the deviation from the "correct" first eigenvector that these occlusions introduce when IVT, ITWVT/R or ITWVT/M is used. As "correct" eigenvectors we consider the eigenvectors at frame 100, with a forgetting factor fixed to 1.0 and the temporal weights up to frame 100 equal to 1.0. Note that the first eigenvector is the one with the highest eigenvalue, and therefore the most important one for computing particle weights (13). The deviation is computed as the distance between the first "correct" eigenvector and the first eigenvectors given by each algorithm. In Fig. 6, a plot of these distances is given, showing that ITWVT/R and ITWVT/M keep the eigenvectors closer to those before the occlusions start. As commented before, ITWVT/M produces smaller sample weights (Fig. 7), which makes the distances slightly smaller than with ITWVT/R.

Finally, to show the polyvalence of the algorithms presented here, we have performed several experiments in two other tracking applications: pedestrian tracking and vehicle tracking. The videos present very deformable objects, in the case of pedestrian tracking, and very hard illumination conditions, in the case of vehicle tracking, but they do not present particular difficulties in terms of partial occlusions, which makes that similar performances are obtained using ITWVT and ITWVTSP. Here we show the results with ITWVTSP/R-iso.

In Fig. 8, the tracking of a subject in sequence S1-T1-C of Camera 3 of the PETS2006 Dataset[3] is shown. Given the variability on the appearance of a pedestrian, mainly due to the legs, we use an "iso" spatial weighting strategy but with the Gaussian shape displaced toward the upper part of the patch. This gives more importance to the body of the pedestrian than to his legs. The maximum spatial weight used is ${}^{s}\omega_{\max} = 3.2$ and the noise threshold $\varepsilon = 0.12$. We use the higher boundary of the values of ε proposed in Section IV-A due to the high deformability of the tracked object.

V. CONCLUSION AND PERSPECTIVES

In this paper, we introduced an incremental PCA algorithm with weighted samples, the incremental temporally weighted PCA (ITWPCA) algorithm. This algorithm can be used in any

[3]Available at http://www.cvg.reading.ac.uk/PETS2006/data.html (last visited in May 2012).

application requiring an incremental computation of a PCA, due to either computational requirements or the lack of the whole dataset at the beginning. The capacity of this algorithm for weighting the contribution of samples can be used for minimizing the impact of outliers in the computed PCA. Using this algorithm, a robust VT algorithm capable of being constantly adapted to the tracked object, while trying to avoid model drift, was also developed: the ITWVTSP algorithm. This algorithm considered temporal weights for penalizing bad quality samples added to the object representation model, and spatial weights for giving more importance to some predefined regions of the tracked object. Note that, in the paper, we denoted by ITWVT, the ITWVTSP algorithm with the spatial weights fixed to 1. The combination of these two weighting strategies produced an improvement in terms of RMSE values on the test sequences of around 26% (Table IV). When comparing precision and lost track ratio scores, the increase of the robustness given by these weighting strategies is also clearly noticeable (Table V).

Several alternatives for the computation of temporal weights and spatial penalty were introduced, producing a family of VT algorithms. All the alternatives were tested on challenging video sequences, showing their good performance compared to state-of-the-art techniques, and their polyvalence with respect to the scenario of application. Indeed, the algorithms were applied to face tracking, pedestrian tracking, vehicle tracking, and on the tracking of a rigid and static object (the poster). On video sequences where the tracked object was labeled, the superiority of the proposed approach against state-of-the-art techniques was clearly shown by means of RMSE, precision, and lost track ratio. The performed tests also allowed defining intervals for the values of the parameters that managed the proposed algorithms.

ITWVTSP considers two weighting strategies: the temporal weighting of samples and the spatial penalty of hypothesis. With respect to the temporal weighting, a more in-deep interaction between the particle filter and the weighting strategy arises as an interesting future line of research to be explored. Indeed, the weights of the particles seemed to be a good source of information about the quality of the tracking and therefore could be used for modulating the contribution of samples to the PCA. With respect to spatial weights, in the *Rockstar* sequence we saw that changes in the appearance of the tracked object in spatially important regions can decrease the performance of ITWVTSP compared to ITWVT. This suggested the study of dynamical spatial penalty strategies. Indeed, reconstruction error gave valuable spatial information about changes of the object appearance. This information could be used for adapting dynamically the values of the spatial weights.

In Figs. 9 and 10, a vehicle tracking is performed. In the first sequence, the tracked vehicle experiences extreme and sudden changes in its illumination, which can be observed in the temporal weights going to zero. In the second sequence, recorded at night, the illumination is considerably bad during the whole sequence, but without any significant variation of the conditions. This can also be observed in the weights, which are around the same values during the whole sequence.

Acknowledgment

The authors would like to acknowledge the three unknown referees for their comments and suggestions, which improved the quality of this paper.

References

[1] E. Polat, M. Yeasin, and R. Sharma, "Robust tracking of human body parts for collaborative human computer interaction," *Comput. Vision Image Understanding*, vol. 89, no. 1, pp. 44–69, 2003.

[2] A. D. Santis and D. Iacoviello, "Robust real time eye tracking for computer interface for disabled people," *Comput. Methods Programs Biomed.*, vol. 96, no. 1, pp. 1–11, 2009.

[3] J.-H. An and K.-S. Hong, "Finger gesture-based mobile user interface using a rear-facing camera," in *Proc. IEEE ICCE*, Jan. 2011, pp. 303–304.

[4] P. Reinartz, M. Lachaise, E. Schmeer, T. Krauss, and H. Runge, "Traffic monitoring with serial images from airborne cameras," *ISPRS J. Photogrammetry Remote Sensing*, vol. 61, nos. 3–4, pp. 149–158, 2006.

[5] T. Semertzidis, K. Dimitropoulos, A. Koutsia, and N. Grammalidis, "Video sensor network for real-time traffic monitoring and surveillance," *Intell. Transport Syst.*, vol. 4, no. 2, pp. 103 –112, Jun. 2010.

[6] K. Huang, L. Wang, T. Tan, and S. Maybank, "A real-time object detecting and tracking system for outdoor night surveillance," *Pattern Recognit.*, vol. 41, no. 1, pp. 432–444, 2008.

[7] M. Baseggio, A. Cenedese, P. Merlo, M. Pozzi, and L. Schenato, "Distributed perimeter patrolling and tracking for camera networks," in *Proc. 49th IEEE CDC*, Dec. 2010, pp. 2093–2098.

[8] D. Marimon and T. Ebrahimi, "Combination of video-based camera trackers using a dynamically adapted particle filter," in *Proc. 2nd Int. Conf. Comput. Vision Theory Appl. (VISAPP)*, 2007, pp. 363–370.

[9] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, article 13, Dec. 2006.

[10] E. Maggio and A. Cavallaro, *Video Tracking: Theory and Practice*. New York, USA: Wiley, 2010.

[11] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Int. Joint Conf. Artif. Intell.*, 1981, pp. 674–679.

[12] C. Tomasi and T. Kanade, "Detection and tracking of point features," *Int. J. Comput. Vision*, Tech. Rep. CMU-CS-91-132, Apr. 1991.

[13] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–577, May 2003.

[14] M. A. Haj, A. Bagdanov, J. Gonzàlez, and F. Roca, "Reactive object tracking with a single PTZ camera," in *Proc. 20th ICPR*, Aug. 2010, pp. 1690–1693.

[15] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-n learning: Bootstrapping binary classifiers by structural constraints," in *Proc. IEEE CVPR*, Jun. 2010, pp. 49–56.

[16] M. Isard and A. Blake, "CONDENSATION conditional density propagation for visual tracking," *Int. J. Comput. Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[17] S. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Trans. Image Process.*, vol. 13, no. 11, pp. 1491–1506, Nov. 2004.

[18] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vision*, vol. 77, nos. 1–3, pp. 125–141, 2008.

[19] E. Maggio, M. Taj, and A. Cavallaro, "Efficient multitarget visual tracking using random finite sets," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 8, pp. 1016–1027, Aug. 2008.

[20] Y. Lao, J. Zhu, and Y. Zheng, "Sequential particle generation for visual tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 9, pp. 1365–1378, Sep. 2009.

[21] T. Cootes, G. Edwards, and C. Taylor, "Active appearance models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 681–685, Jun. 2001.

[22] K.-C. Lee, J. Ho, M.-H. Yang, and D. Kriegman, "Visual tracking and recognition using probabilistic appearance manifolds," *Comput. Vision Image Understanding*, vol. 99, no. 3, pp. 303–331, 2005.

[23] C. Stauffer and W. Grimson, "Adaptive background mixture models for real-time tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, vol. 2. Jun. 1999, pp. 637–663.

[24] V. Papadourakis and A. Argyros, "Multiple objects tracking in the presence of long-term occlusions," *Comput. Vision Image Understanding*, vol. 114, no. 7, pp. 835–846, 2010.

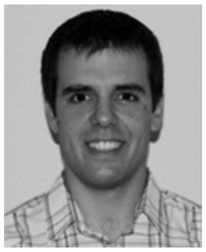
[25] S. Birchfield and S. Rangarajan, "Spatiograms versus histograms for region-based tracking," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit.*, vol. 2. Jun. 2005, pp. 1158–1163.

[26] N.-S. Peng, J. Yang, and Z. Liu, "Mean shift blob tracking with kernel histogram filtering and hypothesis testing," *Pattern Recognit. Lett.*, vol. 26, no. 5, pp. 605–614, 2005.

[27] S. Park and J. K. Aggarwal, "A hierarchical Bayesian network for event recognition of human actions and interactions," *Multimedia Syst.*, vol. 10, no. 2, pp. 164–179, 2004.

[28] X. Liu, L. Lin, S. Yan, H. Jin, and W. Jiang, "Adaptive object tracking by learning hybrid template online," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 11, pp. 1588–1599, Nov. 2011.

[29] V. Ablavsky and S. Sclaroff, "Layered graphical models for tracking partially occluded objects," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 9, pp. 1758–1775, Sep. 2011.
[30] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via on-line boosting," in *Proc. BMVC*, vol. 1. 2006, pp. 47–56.
[31] Y. Iwahori, N. Enda, S. Fukui, H. Kawanaka, R. Woodham, and Y. Adachi, "Efficient tracking with adaboost and particle filter under complicated background," in *Knowledge-Based Intelligent Information and Engineering Systems* (Lecture Notes in Computer Science, vol. 5178), I. Lovrek, R. Howlett, and L. Jain, Eds. Berlin/Heidelberg, Germany: Springer, 2008, pp. 887–894.
[32] O. Williams, A. Blake, and R. Cipolla, "Sparse Bayesian learning for efficient visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 8, pp. 1292–1304, Aug. 2005.
[33] X. Mei and H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 11, pp. 2259–2272, Nov. 2011.
[34] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosoph. Mag.*, vol. 2, no. 6, pp. 559–572, 1901.
[35] P. Hall, D. Marshall, and R. Martin, "Incremental eigenanalysis for classification," in *Proc. Brit. Mach. Vision Conf.*, 1998, pp. 286–295.
[36] A. Levy and M. Lindenbaum, "Sequential Karhunen–Loeve basis extraction and its application to images," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1371–1374, Aug. 2000.
[37] M. Brand, "Fast low-rank modifications of the thin singular value decomposition," *Linear Algebra Its Appl.*, vol. 415, no. 1, pp. 20–30, 2006.
[38] D. Huang, Z. Yi, and X. Pu, "A new incremental pca algorithm with application to visual learning and recognition," *Neural Process. Lett.*, vol. 30, no. 3, pp. 171–185, Dec. 2009.
[39] D. A. Jackson and Y. Chen, "Robust principal component analysis and outlier detection with ecological data," *Environmetrics*, vol. 15, no. 2, pp. 129–139, 2004.
[40] M. Hubert, P. J. Rousseeuw, and K. V. Branden, "ROBPCA: A new approach to robust principal component analysis," *Technometrics*, vol. 47, pp. 64–79, Feb. 2005.
[41] M. Hubert, P. J. Rousseeuw, and T. Verdonck, "Robust PCA for skewed data and its outlier map," *Comput. Statist. Data Anal.*, vol. 53, no. 6, pp. 2264–2274, Apr. 2009.
[42] H. Kriegel, P. Kriegelöger, E. Schubert, and A. Zimek, "A general framework for increasing the robustness of PCA-based correlation clustering algorithms," in *Proc. 20th Int. Conf. Sci. Statistical Database Manage.*, 2008, pp. 418–435.
[43] D. Skocaj, A. Leonardis, and H. Bischof, "Weighted and robust learning of subspace representations," *Pattern Recognit.*, vol. 40, no. 5, pp. 1556–1569, 2007.
[44] D. Skočaj and A. Leonardis, "Incremental and robust learning of subspace representations," *Image Vision Comput.*, vol. 26, pp. 27–38, Jan. 2008.
[45] J. Czyz, B. Ristic, and B. Macq, "A particle filter for joint detection and tracking of color objects," *Image Vision Comput.*, vol. 25, no. 8, pp. 1271–1281, 2007.
[46] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *Proc. 12th Int. Conf. Comput. Vision*, Oct. 2009, pp. 1515–1522.
[47] M.-H. Yang, D. Kriegman, and N. Ahuja, "Detecting faces in images: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 1, pp. 34–58, Jan. 2002.
[48] J. Mundy, "Object recognition in the geometric era: A retrospective," in *Toward Category-Level Object Recognition* (Lecture Notes in Computer Science, vol. 4170), J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, Eds. Berlin/Heidelberg, Germany: Springer, 2006, pp. 3–28.
[49] M. Enzweiler and D. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2179–2195, Dec. 2009.
[50] C. Galleguillos and S. Belongie, "Context based object categorization: A critical survey," *Comput. Vision Image Understanding*, vol. 114, no. 6, pp. 712–722, 2010.
[51] D. Gerónimo, A. López, A. Sappa, and T. Graf, "Survey of pedestrian detection for advanced driver assistance systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1239–1258, Jul. 2010.
[52] J. Cruz-Mota, "Model-based behavioural tracking and scale invariant features in omnidirectional matching," Ph.D. dissertation, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 2011.
[53] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *J. Roy. Statist. Soc. B*, vol. 61, pp. 611–622, Sep. 1999.
[54] A. Jepson, D. Fleet, and T. El-Maraghi, "Robust online appearance models for visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 10, pp. 1296–1311, Oct. 2003.



Javier Cruz-Mota was born in Barcelona, Spain, in 1982. He received the Mathematics and Telecommunications Engineering degrees from the Universitat Politècnica de Catalunya, Barcelona, Spain, in 2005 and 2006, respectively, and the Ph.D. degree in Electrical Engineering from the École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, in 2011.

He is currently a Research and Teaching Assistant at the Transport and Mobility Laboratory, EPFL. His current research interests include image and signal processing and computer vision, in particular, visual tracking, mathematical modeling and simulation, omnidirectional vision, and behavioral information in image analysis.

Michel Bierlaire was born in Belgium, in 1967. He received the Ph.D. degree in mathematical sciences from the Facultés Universitaires Notre-Dame de la Paix, University of Namur, Namur, Belgium.

Between 1995 and 1998, he was a Research Associate and the Project Manager at the Intelligent Transportation Systems Program, Massachusetts Institute of Technology, Cambridge, MA, USA. Between 1998 and 2006, he was a Junior Faculty Member at the Operations Research Group ROSO, Institute of Mathematics, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland. In 2006, he was an Associate Professor at the School of Architecture, Civil, and Environmental Engineering, EPFL, where he became the Director of the Transport and Mobility Laboratory. Since 2009, he has been the Director of TraCE, the Transportation Center, and the Director of the Doctoral Program in Civil and Environmental Engineering, EPFL. His main expertise is in the design, development, and applications of models and algorithms for the design, analysis, and management of transportation systems. Namely, he has been active in demand modeling (discrete choice models, estimation of origin-destination matrices) and dynamic traffic management systems.

Jean-Philippe Thiran (M'93–SM'05) was born in Namur, Belgium, in 1970. He received the Elect. Eng. and Ph.D. degrees from the Université Catholique de Louvain, Louvain-la-Neuve, Belgium, in 1993 and 1997, respectively.

He joined the École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland, in February 1998 as a Senior Lecturer, responsible for the Image Analysis Group. Since 2004, he has been the Director of the EPFL Signal Processing Laboratory (LTS5), and in 2011, he became an Associate Professor of signal processing at EPFL. He is a part-time Associate Professor at the School of Medicine, University of Lausanne, Lausanne. He is an author or co-author of one book, nine book chapters, 90 journal papers, and more than 150 peer-reviewed papers published in proceedings of international conferences. He holds four international patents. His current research interests include image segmentation, prior knowledge integration in image analysis, partial differential equations, and variational methods in image analysis, multimodal signal processing, medical image analysis, including multimodal image registration, segmentation, computer-assisted surgery, and diffusion MRI.

Dr. Thiran was the Co-Editor-in-Chief of the *Signal Processing* journal (published by Elsevier Science) from 2001 to 2005. He is currently an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, an Associate Editor of the *International Journal of Image and Video Processing*, and a member of the Editorial Board of the *Signal, Image and Video Processing* journal. Among many other scientific duties, he was the General Chairman of the 2008 European Signal Processing Conference, the Tutorial Co-Chair of the IEEE International Conference on Image Processing (ICIP) in 2011, and will be the Technical Co-Chair of ICIP 2015.