Sample and Pixel Weighting Strategies for Robust Incremental Visual Tracking

Javier Cruz-Mota * Michel Bierlaire *
Jean-Philippe Thiran †

October 27, 2011

Report TRANSP-OR 111027
Transport and Mobility Laboratory
Ecole Polytechnique Fédérale de Lausanne
transp-or.epfl.ch

*Transp-OR, École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland, javier.cruz@epfl.ch, michel.bierlaire@epfl.ch

†LTS5, École Polytechnique Fédérale de Lausanne, CH-1015 Lausanne, Switzerland, jp.thiran@epfl.ch

Abstract

In this paper we introduce the Incremental Temporally Weighted Principal Component Analysis (ITWPCA) algorithm, based on SVD 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, in order 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 penalising 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.

1 Introduction

Visual Tracking (VT) is a core problem in many Computer Vision (CV) applications, such as Human-Computer Interaction (HCI) (Polat et al., 2003; Santis and Iacoviello, 2009; An and Hong, 2011), traffic monitoring (Reinartz et al., 2006; Semertzidis et al., 2010), video-surveillance (Huang et al., 2008; Baseggio et al., 2010) or Augmented Reality (AR) (Marimon and Ebrahimi, 2007). 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 composed of image frames I_k, i.e.

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

where k is a temporal index, a tracking algorithm estimates for every tracked object j, a time series

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

$j \in J$, 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 (Yilmaz et al., 2006; Maggio and Cavallaro, 2010): the feature extraction block, the object representation block and the object localisation block. A generic VT algorithm can be seen as the application of these three blocks according to the schematic representation in Figure 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, i.e. the space where the characteristics of the tracked object will be defined, such as colour, motion, edges or interest points. The object localisation block takes information from the object representation model and the features to estimate the new state of the object of interest. In general, localisation is performed under the hypothesis of a smooth change of position, shape and appearance. Object localisation algorithms can compute the target state analytically, as the solution of an optimisation problem where a cost function is minimised (Lucas and Kanade, 1981; Tomasi and Kanade, 1991; Comaniciu et al., 2003; Haj et al., 2010; Kalal et al., 2010), or by evaluating simultaneously multiple candidate tracks (hypothesis)

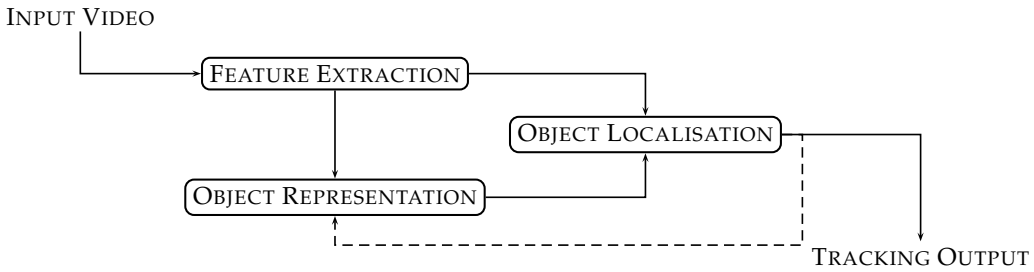


Figure 1: Schematic view of a general VT algorithm.

per object of interest per time step using Particle Filters (PF) (Isard and Blake, 1998; Zhou et al., 2004; Ross et al., 2008). PF validate these hypothesis against visual information and motion models and their main advantage is their capacity to deal with multi-modality and therefore with clutter.

All these object localisation 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 them, such as Principal Component Analysis (PCA) (Cootes et al., 2001; Lee et al., 2005; Ross et al., 2008), mixtures of Gaussians (Stauffer and Grimson, 1999; Papadourakis and Argyros, 2010), histograms (Birchfield and Rangarajan, 2005; Peng et al., 2005), bayesian networks (Park and Aggarwal, 2004) or boosting techniques (Grabner et al., 2006; Iwahori et al., 2008). 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 initialisation 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 makes this strategy more effective in terms of adaptability, since the type of changes that the model has to handle does not need to be known beforehand. However, the 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.

PCA is a well-known and commonly used technique for dimensionality reduction (Pearson, 1901), usually employed for computing object representation models. 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 (Hall et al., 1998; Levy and Lindenbaum, 2000; Brand, 2006; Ross et al., 2008). This way, instead of computing a big PCA on a big data matrix, a PCA is performed on a small sub-matrix. 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 (Levy and Lindenbaum, 2000), the sequential computation of PCA is tackled by updating an existing PCA with the components of the new data that are orthogonal to the previously generated subspace. 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 factorisation 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 computationally efficient than updating the PCA for every new data sample, which makes methods based on (Levy and Lindenbaum, 2000) more efficient than those based on (Hall et al., 1998) (see (Huang et al., 2009) for comparison details).

With respect to outliers, two strategies arise in a PCA computation procedure for minimising their effect on principal components. The first option is to discard samples that are supposed to be outliers. This forces to have a good outlier detection approach, in order to do not discard good samples. For instance, in (Jackson and Chen, 2004) a minimum volume ellipsoid is fitted to data in order to discard, in the PCA computation, the samples that are outside; and in (Hubert et al., 2005) and (Hubert et al., 2009), samples are discarded according to their projection in a subspace computed using a robust covariance estimation. The second option is to weight the contribution of each value of the data matrix according to a measure of confidence. In this kind of approaches, the PCA computation is still dependent on the outlier detection method, but its depen-

dency is considerably weakened since all received data is considered. In (Kriegel et al., 2008), the authors 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 (Skocaj et al., 2007), 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 (Skočaj and Leonardis, 2008), an incremental weighted PCA algorithm is introduced. The drawback of this algorithm is that it is based on the incremental PCA algorithm introduced in (Hall et al., 1998) 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.

In this paper we introduce an incremental PCA algorithm with temporal weights, the Incremental Temporally Weighted Principal Component Analysis (ITWPCA) algorithm. It is based on SVD update (Levy and Lindenbaum, 2000) and therefore can compute the incremental step using blocks of new data instead of individual samples, which makes the algorithm more computationally efficient. Then, a robust VT algorithm based on a particle filter approach and the ITWPCA algorithm for object representation computation is also introduced. This VT algorithm computes an object representation model of the tracked object using a PCA on rectangular templates. The use of the ITWPCA algorithm for computing the PCA allows, on the one hand, to maintain the object representation model constantly adapted to the tracked object, and on the other hand to reduce the impact on the PCA of bad quality samples of the target. 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 to assign more importance to some predefined regions of the tracked object, producing a higher accuracy on the tracking.

Let us note that a common preliminary operation to tracking is object detection, although some algorithms perform a simultaneous detection and tracking (Czyz et al., 2007; Breitenstein et al., 2009). 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 (Yang et al., 2002; Mundy, 2006; Enzweiler and Gavrila, 2009; Galleguillos and Belongie, 2010; Gerónimo et al., 2010) and references inside them.

This paper is organised as follows. In Section 2, an incremental PCA algorithm that considers temporal weights is introduced. Then, in Section 3, a VT algorithm that employs this PCA algorithm for updating the object representation model is described in detail. This algorithm also considers spatial weights that can increase the importance of a region of the target, increasing the tracking accuracy. In Section 4, 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 5 conclusions and future lines of research are derived.

2 Incremental PCA with Weighted Samples: the ITWPCA Algorithm

In (Levy and Lindenbaum, 2000), 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, with respect to the subspace generated by the PCA matrix, of the new data. In (Ross et al., 2008), 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 minimises the weighted squared reconstruction error

$$\widetilde{\xi} = \sum_{i=1}^{M} \sum_{j=1}^{N} \omega_{ij} \left(\widehat{z}_{ij} - \sum_{p=1}^{K} u_{ip} \sum_{q=1}^{M} u_{qp} \widehat{z}_{qj} \right)^2 . \tag{3}$$

where a_{bc} represents the element at row b and column c of matrix A, and $\widehat{Z}$ is the matrix obtained by subtracting the temporally weighted mean to each column of Z. The elements of the temporally weighted mean vector,

μ_i, are computed as

$$\mu_i = \frac{1}{\sum_{j=1}^{N} \omega_{ij}} \sum_{j=1}^{N} \omega_{ij} z_j, \tag{4}$$

and so $\mu = [\mu_1, \ldots, \mu_M]^\top$.

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

$$\widetilde{\xi} = \sum_{i=1}^{M} \sum_{j=1}^{N} \left(\widetilde{\widehat{z}}_{ij} - \sum_{p=1}^{K} u_{ip} \sum_{q=1}^{M} u_{qp} \widetilde{\widehat{z}}_{qj} \right)^2, \tag{5}$$

where $\widetilde{\widehat{z}}_{ij} = \sqrt{\omega_j}\widehat{z}_{ij}$. Then, the matrix U that minimises $\widetilde{\xi}$ is composed by the K biggest eigenvalues of the covariance matrix of $\widetilde{\widehat{Z}}$, and can be computed by performing Singular Value Decomposition on this matrix, i.e. SVD$(\widetilde{\widehat{Z}}) = U\Sigma V^\top$, as introduced in (Skocaj et al., 2007).

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 multiple, equal to $\sum_{i=1}^{N} \omega_i$. Therefore, eigenvectors of both matrices are the same and eigenvalues are scaled by this scalar multiple. This makes equivalent to work with the covariance matrix or the scatter matrix, in terms of PCA. Let us now introduce the following lemma:

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 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 is simple and can be found in the Appendix. 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 sub-matrices. Basically, the new scatter matrix is the sum of the other two, plus 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 (Ross et al., 2008) can be adapted to consider temporal weights as expressed in the Incremental Temporally Weighted PCA (ITWPCA) algorithm described in Algorithm 1.

3 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 changes along time on the appearance of the tracked object. In (Ross et al., 2008), the authors 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 Incremental Temporally Weighted Visual Tracking with Spatial Penalty (ITWVTSP) algorithm. This VT algorithm exploits the capacity of the ITWPCA algorithm (introduced in Section 2) 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 to decrease the impact on the PCA of bad quality tracked samples, which reduces the risk of model drift and makes the

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 $\mu^{(2)} = \frac{1}{\|{}^t\omega^{(2)}\|_1} \sum_{i=1}^{N^{(2)}} \omega_i^{(2)} z_i^{(2)}$ 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: Compute $\widetilde{\widehat{Z}}^{(2)} = [\sqrt{\omega_1^{(2)}}(x_1^{(2)} - \mu^{(2)}), \ldots, \sqrt{\omega_{N^{(2)}}^{(2)}}(x_{N^{(2)}}^{(2)} - \mu^{(2)}), \sqrt{\frac{\|{}^t\omega^{(1)}\|_1 \|{}^t\omega^{(2)}\|_1}{\|{}^t\omega^{(1)}\|_1 + \|{}^t\omega^{(2)}\|_1}}(\mu^{(1)} - \mu^{(2)})]$

3: Compute ${}^{\perp}\widetilde{\widehat{Z}}^{(2)} = orth(\widetilde{\widehat{Z}}^{(2)} - U^{(1)}U^{(1)\top}\widetilde{\widehat{Z}}^{(2)})$

4: Compute $R = \begin{bmatrix} f\Sigma^{(1)} & U^{(1)\top}\widetilde{\widehat{Z}}^{(2)} \\ 0 & {}^{\perp}\widetilde{\widehat{Z}}^{(2)\top}(\widetilde{\widehat{Z}}^{(2)} - U^{(1)}U^{(1)\top}\widetilde{\widehat{Z}}^{(2)}) \end{bmatrix}$

5: Compute $SVD(R) = U'\Sigma'V'^{\top}$

6: Then, $U^{(1,2)} = [U^{(1)}\ {}^{\perp}\widetilde{\widehat{Z}}^{(2)}]U'$ and $\Sigma^{(1,2)} = \Sigma'$.

tracking more robust. Furthermore, we introduce spatial weights in order 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 fix them to one, the Incremental Temporally Weighted Visual Tracking (ITWVT) algorithm is obtained. We start by introducing this algorithm in Section 3.1 for afterwards introduce the "Spatial Penalty" capacity in Section 3.2.

3.1 The Incremental Temporally Weighted Visual Tracking (ITWVT) Algorithm

A probabilistic interpretation of PCA (Tipping and Bishop, 1999) allows to combine the object representation of a target, computed using PCA, with a particle filter approach for object localisation. The most appropriate appearance representations for this setup is a rectangular template, considering as the state-space the six parameters of an affine transformation: translation (2 parameters), rotation angle, scale, aspect ratio and skew direction.

The particles, that are placed in this 6-dimensional 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 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

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

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 (Isard and Blake, 1998) 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 auto-covariance 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 auto-covariance 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 j-th eigenvector. Therefore, the weight of particle i at time step k before normalisation 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 ITW-PCA algorithm is used for updating the PCA matrix. Let us introduce two measures of the quality of a tracked sample. 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 is the new sample from the PCA subspace. Then, let us define the confidence on the tracked patch at time step k, c_k, as

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

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

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

and

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

being $\bar{z}_i$ the i-th component of the vector $\bar{z} = UU^\top(z - \mu)$. The measure proposed in Eq. (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 more than ε error are discarded ($\omega = 0$). This strategy tries to penalise 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 increased, with the risk of being too restrictive and therefore becoming unadapted to the object of interest.

3.2 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 more importance to some pixel than to others, 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 in order 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.

A higher tracking accuracy on this important regions, which translates to a higher tracking accuracy, can be achieved by penalising the contribution of important pixels to the distances in Eqs. (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, since hypothesis fitting better these more penalised pixels will be favoured. Thus, let us redefine Eqs. (10) and (11) by considering spatial weights as

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

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

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

Using these equations for computing particle weights considers the importance given beforehand to every 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 take into account is that an excessive increase of the weight

applied to certain pixels can render the tracking algorithm unstable (as will be shown in Section 4).

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 ${}^s\omega = \mathbb{1}_{M\times 1}$, we obtain the ITWVT algorithm. We denote by "/R" the use of Eq. (15) and by "/M" the use of Eq. (16), i.e. for instance we denote by ITWVT/M the ITWVTSP algorithm using $f_M(z, \varepsilon)$ and ${}^s\omega = \mathbb{1}_{M\times 1}$.

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.

$\mu = z_0$, $n = 1$, and $U^{(1)}$, $\Sigma^{(1)}$, $Z^{(2)}$ and ${}^t\omega^{(2)}$ are empty
Set ${}^s\omega$ to the desired spatial weights (by default, ${}^s\omega = \mathbb{1}_{M\times 1}$)
for every frame of the video **do**
    Draw particles according to the dynamical model (Eq. (8)) and the weight distribution of particles.
    For each particle, compute its weight according to the observation model and spatial weights (Eq. (17) and Eq. (18)).
    Store in $Z^{(2)}$ the image region corresponding to the most likely particle, and in ${}^t\omega^{(2)}$ its PCA weight (Eq. (14))
    if there are $N^{(2)}$ stored images in $Z^{(2)}$ **then**
        if $n < K$ **then**
            ${}^t\omega_i^{(2)} = 1,\ \forall i = 1, \ldots, N^{(2)}$
        Apply Algorithm 1 with $\|{}^t\omega^{(1)}\|_1 = n$, discarding the eigenvectors that exceed K.
        Set $U^{(1)} = U^{(1,2)}$, $\Sigma^{(1)} = \Sigma^{(1,2)}$ and $n = fn + \|{}^t\omega^{(2)}\|_1$
        Empty $Z^{(2)}$ and ${}^t\omega^{(2)}$

4 Tests and Results

We have performed several tests to the ITWVT and the ITWVTSP algorithms on several video sequences. For showing the improvement obtained by the weighting strategy, the results are compared with the results

obtained by the IVT algorithm introduced in (Ross et al., 2008)[1]. For a general comparison against state-of-the-art algorithms, we compare also our results with the results obtained with the TLD algorithm introduced in Kalal et al., 2010[2].

With IVT, ITWVT and ITWVTSP, we use the same parameters than those proposed in (Ross et al., 2008), i.e. 600 particles, an eigenvector size of 32×32 pixels, a maximum number of 16 eigenvectors and a block update of 5 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. With these parameters, the implementation of ITWVTSP in MatLab runs at 7 frames per second in a laptop with a 2.0GHz processor.

The standard deviations of the dynamical model (Eq. (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, which are similar values to those proposed in the implementation of IVT. By using the same parameters for the three algorithms, the performance improvement due to the temporal and spatial weighting strategy can be clearly perceived. With TLD, the standard parameters provided in the distributed implementation are used.

For visualisation of the tracking results, we use the same template as in (Ross et al., 2008): 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. In Figure 2 an example is shown.

The performed experiments are divided in two groups. In the first group, there are experiments performed on labelled 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 unlabelled video sequences in a variety of tracking applications, to show the polyvalence of ITWVT and ITWVTSP.

[1]Implementation available at http://www.cs.toronto.edu/ dross/ivt/ (last visited in october 2011)

[2]Implementation available at http://info.ee.surrey.ac.uk/Personal/Z.Kalal/tld.html (last visited in october 2011)

4.1 Labelled Video Sequences

First, we have performed a tracking of the face in the Dudek sequence (Jepson et al., 2003) (see Figure 2) with IVT, TLD, ITWVT and ITWVTSP. This sequence is a very challenging video with changes in the tracked object, the camera position and the illumination. The ground truth of 7 manually labelled points on the face is available for this sequence. This allows to compare quantitatively the obtained results, by computing the Root Mean Squared Error (RMSE) of the tracked points with respect to the real ones. Given the implicit stochasticity of the algorithms, ten independent runs per algorithm are performed in all the tests. All the runs producing a RMSE bigger than 10.0px are considered as losses of track.

The ITWVT algorithm has been tested using Eq. (15) and Eq. (16), for 25 different values of ε between 0.01 and 0.9. Both variants of the algorithm, ITWVT/M and ITWVT/R, provide similar results, with small variance among runs for $\varepsilon \in [0.02, 0.12]$. 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, and 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.

According to the obtained results, a reasonable value for the error threshold is $\varepsilon = 0.07$. For this value, ITWVT/M obtains a best RMSE $= 5.6537$px and ITWVT/R a best RMSE $= 5.8645$px. The best run with IVT obtains a RMSE $= 6.2324$px, which shows that the use of temporal weights improves the performance of the tracking. In addition, only one out of the ten runs lost the track (RMSE higher than 10.0px) for both, ITWVT/M and ITWVT/R, while five are lost with IVT. This shows the improvement in the robustness of the tracking thanks to the better quality of the model. Performances of ITWVT/R and ITWVT/M are similar, although looking to 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.

Let us note that the value of ε is application-specific. Indeed, the appearance of a rigid object changes slightly, which allows to fix a more restrictive (smaller) ε. On the contrary, a deformable object like for instance

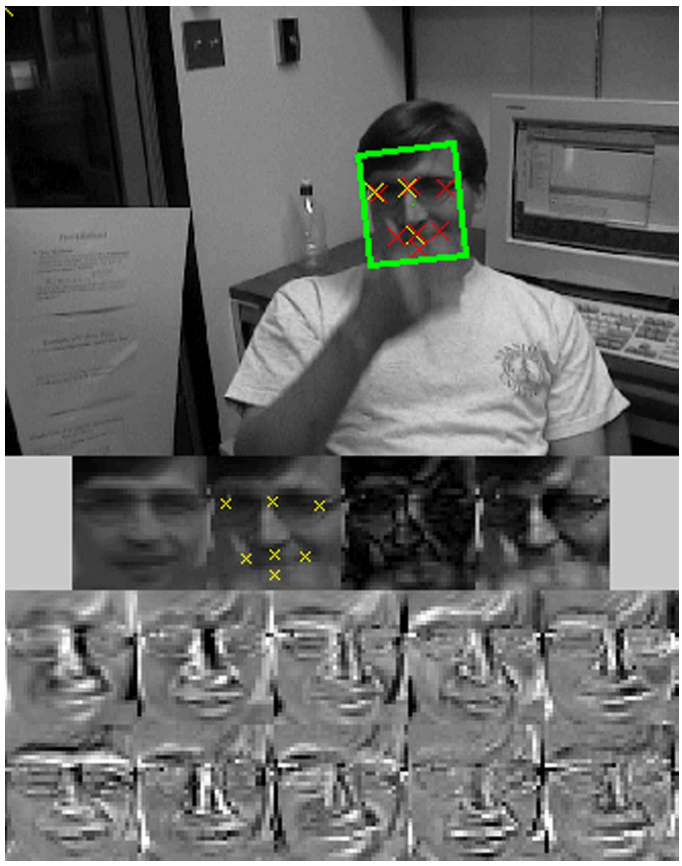
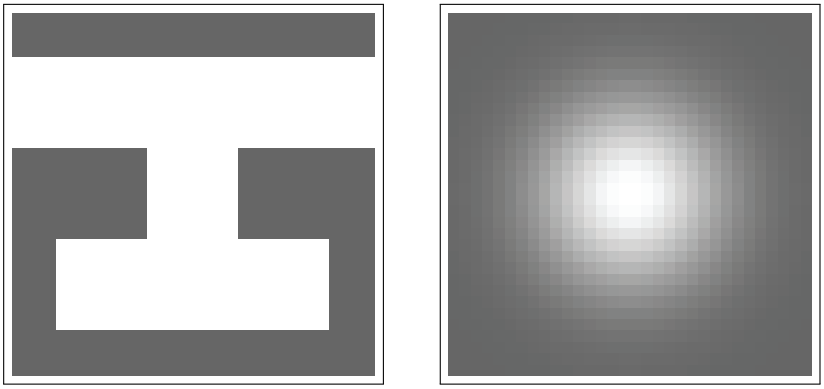


Figure 2: Frame of the Dudek sequence where an occlusion of the face is starting. First row contains the current frame with the tracked region. The second row contains the mean, the tracked window, the reconstruction error and the reconstructed image. Finally, the third and fourth rows contain the first ten eigenvalues.



(a) "Spec" spatial weights (b) "Iso" spatial weights

Figure 3: Spatial weights used in the experiments. Brighter regions correspond to high weight values, darker regions to spatial weights equal to 1.0.

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.

For observing the effect of spatial penalty, we have designed two ${}^s\omega$ vectors. The first one assigns higher weight values to pixels on important regions of the face (see Figure 3(a)), we call this 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 (see Figure 3(b)), we call this the "iso" spatial weights and denote its use by "-iso". The "iso" spatial weights considers that pixels far from the boundary of the tracked object are more important.

For each variant of the algorithm, i.e. ITWVTSP/M-spec, ITWVTSP/M-iso, ITWVTSP/R-spec and ITWVTSP/R-iso, the maximum value of the spatial weight (${}^s\omega_{\text{max}}$) has been varied between 1 and 3.5, launching 10 runs for each value (with ε fixed to $\varepsilon = 0.07$). The minimum value in the ${}^s\omega$ vector is always 1.0. For values of ${}^s\omega_{\text{max}} > 2.0$ 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 make the algorithm more stable allowing to go up

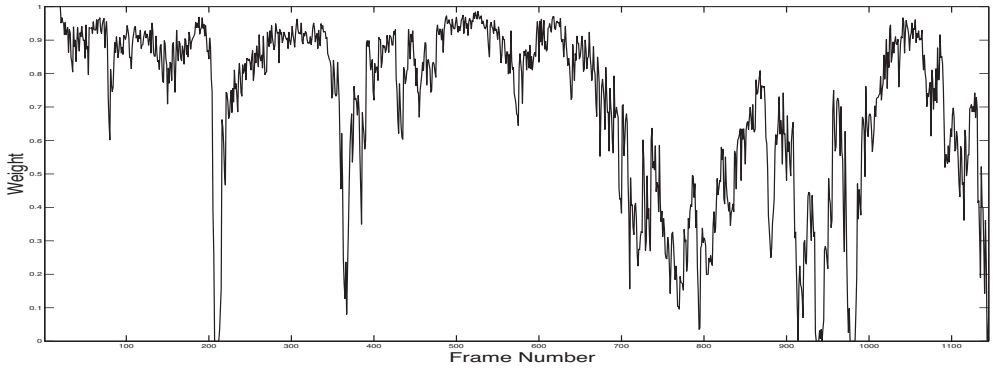


Figure 4: Weights applied to each tracked sample of the Dudek sequence using ITWVTSP/R-Spec with $\varepsilon = 0.07$ and ${}^{s}\omega_{\max} = 1.8$. These weights correspond to the run that gave the best RMSE value (4.5969px, see Table 1). The frames that present an occlusion or the frames where the face is rotated out-of-the-plane are clearly noticeable (small weights).

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.

In Table 1, statistics of values of RMSE obtained with each algorithm are shown, for the parameter values commented above. 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. In this case, this is not due to a loss of track in all the runs. 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 higher RMSE value. In Figure 4, the weights applied to each tracked sample for the best run of ITWVTSP/R-Spec, can be observed.

For testing the algorithms in a real situation with partial occlusions, we recorded the Rockstar sequence. In this sequence, composed of 171 frames, a subject in front of the camera is recorded. At a certain moment, the subject 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

Algorithm	**Losses of Track**	**Mean RMSE**	**Max RMSE**	**Min 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 1: 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".

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, in order to keep eyes, nose and mouth centred along the whole video sequence. This has been done manually for generating the ground truth.

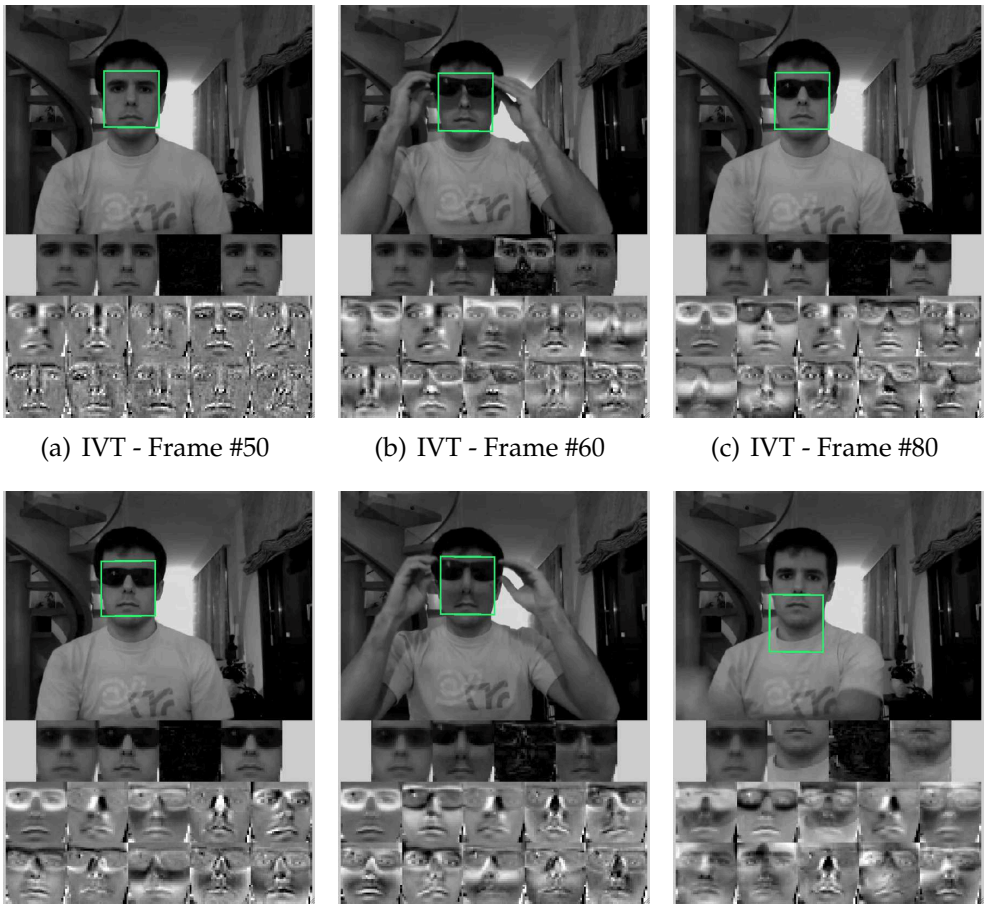
Ten runs of IVT, ITWVT, ITWVTSP and TLD have been performed on this sequence. For spatial weights, a conservative approach has been adopted, taking ${}^s\omega_{\max} = 2.0$ for "iso" and ${}^s\omega_{\max} = 1.6$ for "spec". Precision and lost track ratio scores (Maggio and Cavallaro, 2010) have been computed for all the algorithms and the results are shown in Table 2. For computing 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. 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, in terms of spatial weights, seems to have a negative effect in the performance, 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 Figure 5 and Figure 6, respectively.

4.2 Unlabelled Video Sequences

In Figure 7, 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 the temporal weights, we compute the deviation from the "correct" first eigenvector due to these occlusions in IVT, ITWVT/R and ITWVT/M. 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. The de-

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

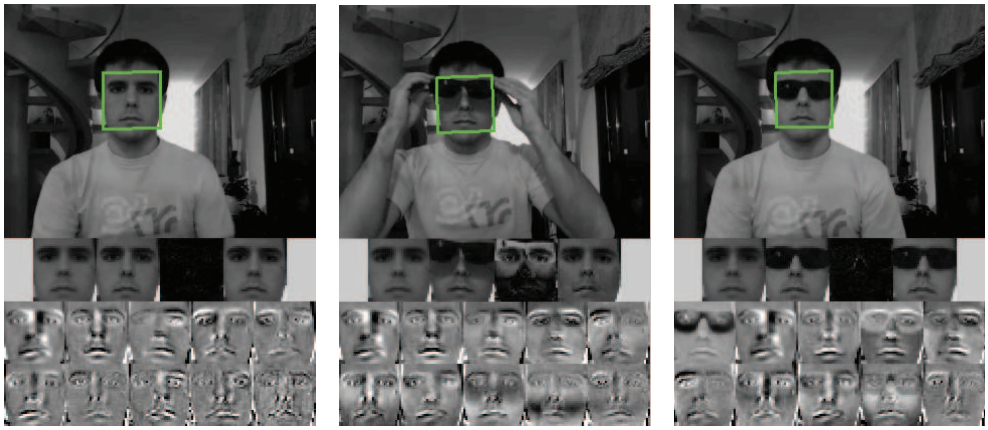
Table 2: Obtained results on the Rockstar sequence. The parameters are $\varepsilon = 0.07$, ${}^{s}\omega_{\max} = 2.0$ for "iso" and ${}^{s}\omega_{\max} = 1.6$ for "spec".



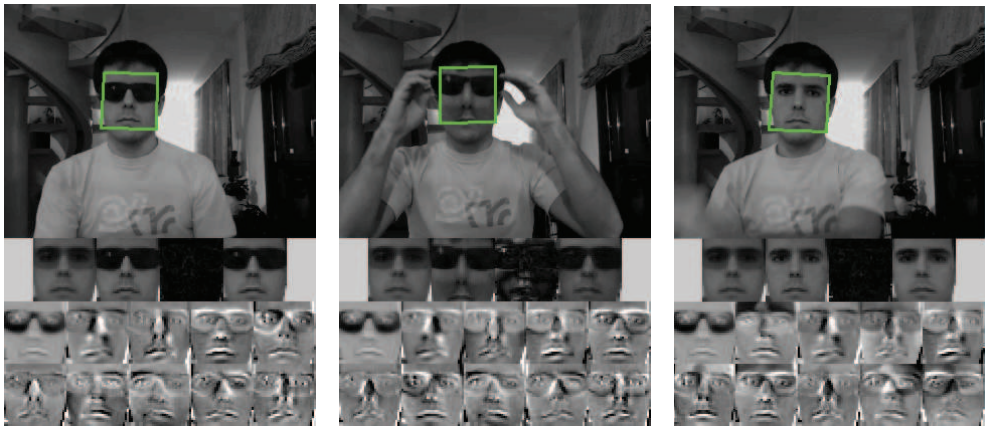
(a) IVT - Frame #50 (b) IVT - Frame #60 (c) IVT - Frame #80

(d) IVT - Frame #115 (e) IVT - Frame #130 (f) IVT - Frame #150

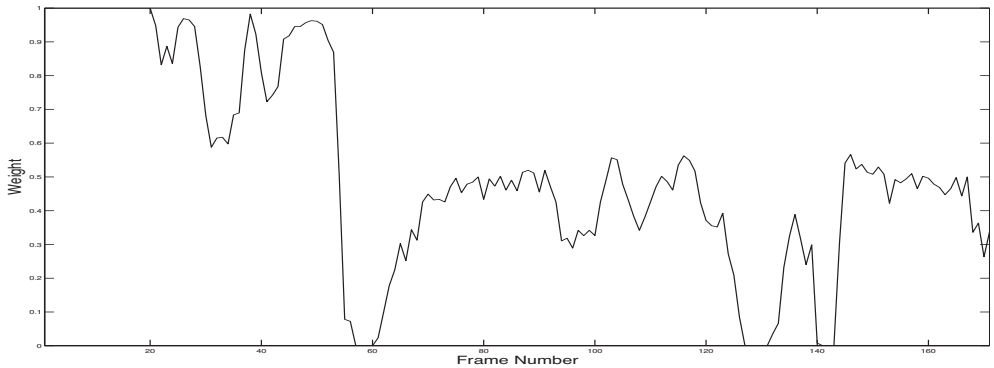
Figure 5: Results obtained with the IVT algorithm on the Rockstar sequence. The eigenvectors show how the sunglasses corrupt the appearance model, avoiding a correct tracking continuation after taking them off.



(a) ITWVT/M - Frame #50 (b) ITWVT/M - Frame #60 (c) ITWVT/M - Frame #80



(d) ITWVT/M - Frame #115 (e) ITWVT/M - Frame #130 (f) ITWVT/M - Frame #150



(g) Weights applied by the ITWVT/M algorithm to every tracked sample of the Rockstar sequence.

Figure 6: Results obtained with the ITWVT/M algorithm on the Rockstar sequence. The eigenvectors show how the information of the sunglasses has a low 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) Frame #100 (b) Frame #161 (c) Frame #310

Figure 7: Several frames of the Poster sequence. The total sequence is composed of 585 frames.

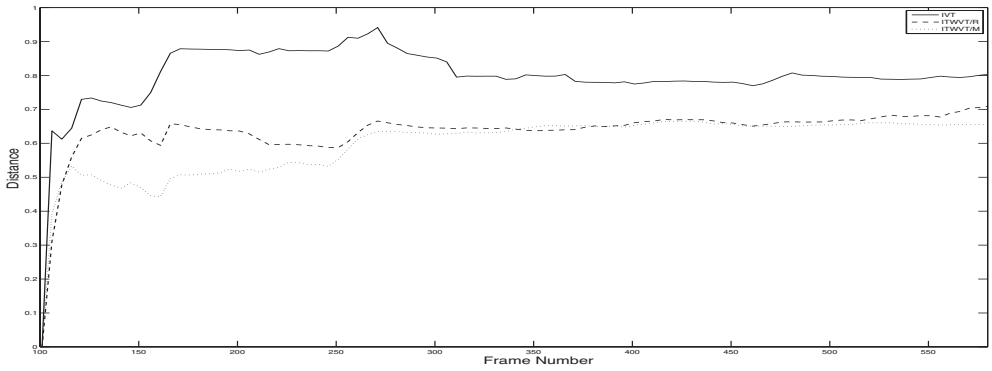
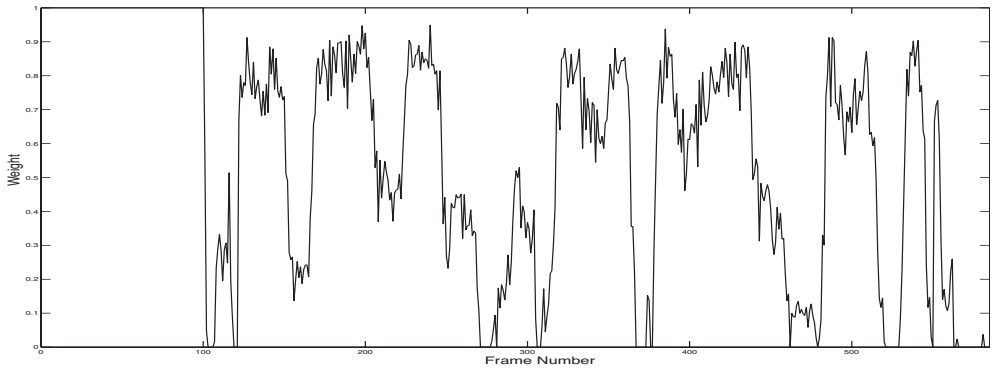


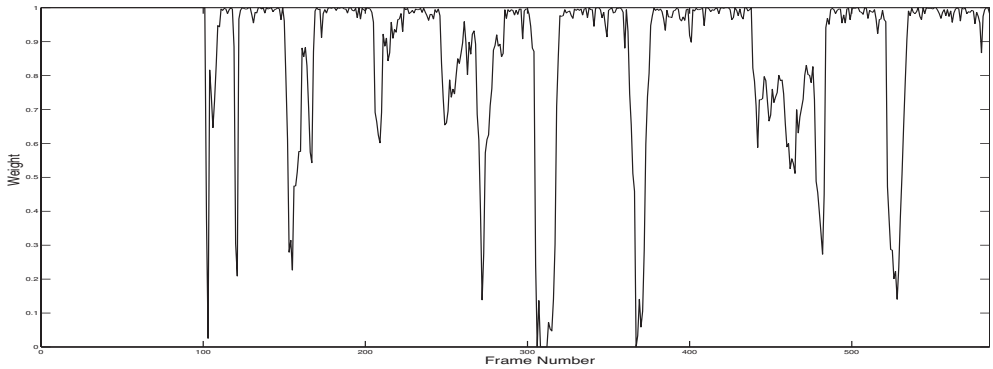
Figure 8: Distances between the first eigenvector at frame 100 and the first eigenvector computed using IVT (solid line), ITWVT/R (dashed line) and ITWVT/M (dotted line) at subsequent frames.

viation is computed as the distance between the first "correct" eigenvector and the first eigenvectors given by each algorithm. In Figure 8, 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 (see Figure 9), 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 do not present particular difficulties in terms of partial occlusions, which makes that similar performances are obtained using ITWVT and ITWVTSP. Here we show



(a) ITWVT/M



(b) ITWVT/R

Figure 9: Weights applied to the samples of the poster in the Poster sequence using ITWVT/M and ITWVT/R.

the results with ITWVTSP/R-iso.

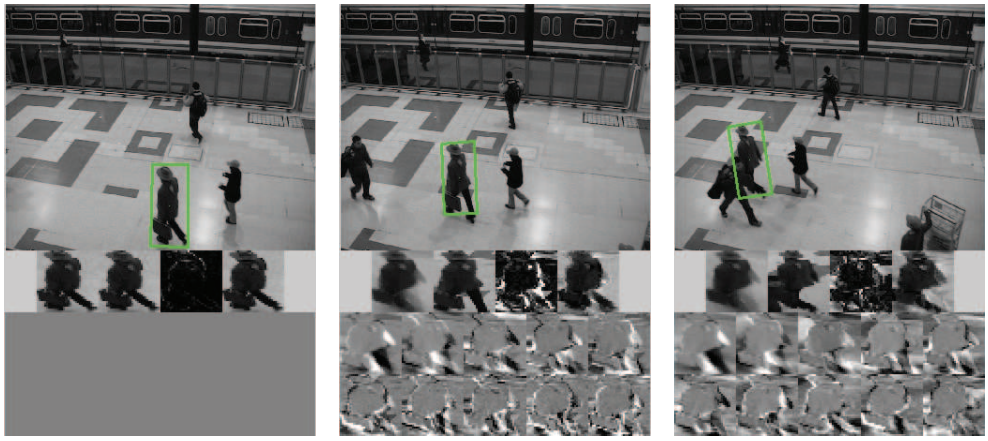
In Figure 10, the tracking of a subject in sequence S1-T1-C of Camera 3 of the PETS2006 Dataset[3] is shown. Given the variability on the appearance of a pedestrian, mainly due to the legs, we use an "iso" spatial weighting strategy but with the Gaussian shape displaced toward the upper part of the patch. 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 $\varepsilon = 0.12$.

In Figure 11 and Figure 12, 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, which runs 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.

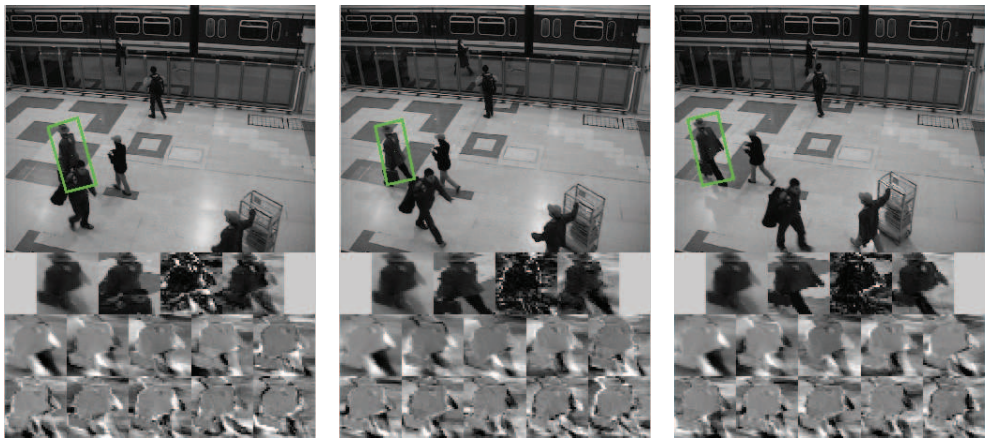
5 Conclusions and Perspectives

In this paper we have introduced an incremental PCA algorithm with weighted samples, the Incremental Temporally Weighted PCA (ITWPCA) algorithm. This algorithm can be used in any 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 minimising 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 has been also developed, the Incremental Temporally Weighted Visual Tracking with Spatial Penalty (ITWVTSP) algorithm. Furthermore, this algorithm allows to consider spatial weights for giving more importance to some regions of the tracked object, which increases tracking accuracy. The combination of these two weighting strategies produce an improvement in terms of RMSE values on the test sequences of around 26% (see Table 1). When comparing Precision and Lost Track Ratio scores, the increase of the robustness given by these weighting strategies is also clearly noticeable (see Table 2).

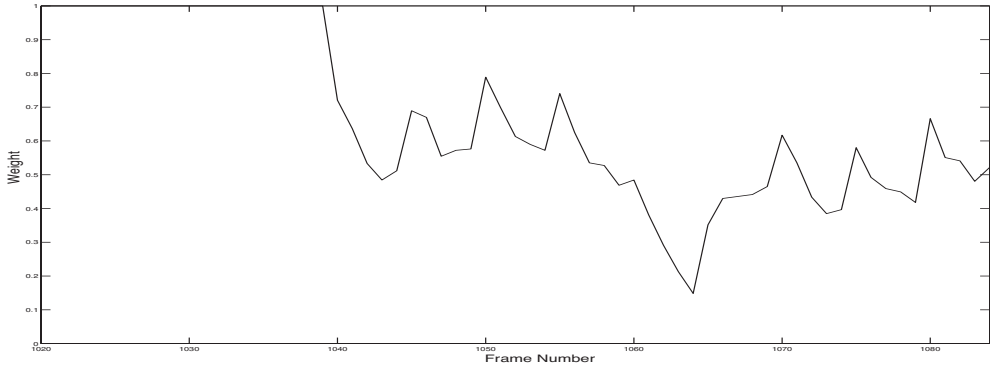
[3]Available at http://www.cvg.reading.ac.uk/PETS2006/data.html (last visited in october 2011)



(a) Frame #1020 (b) Frame #1039 (c) Frame #1059

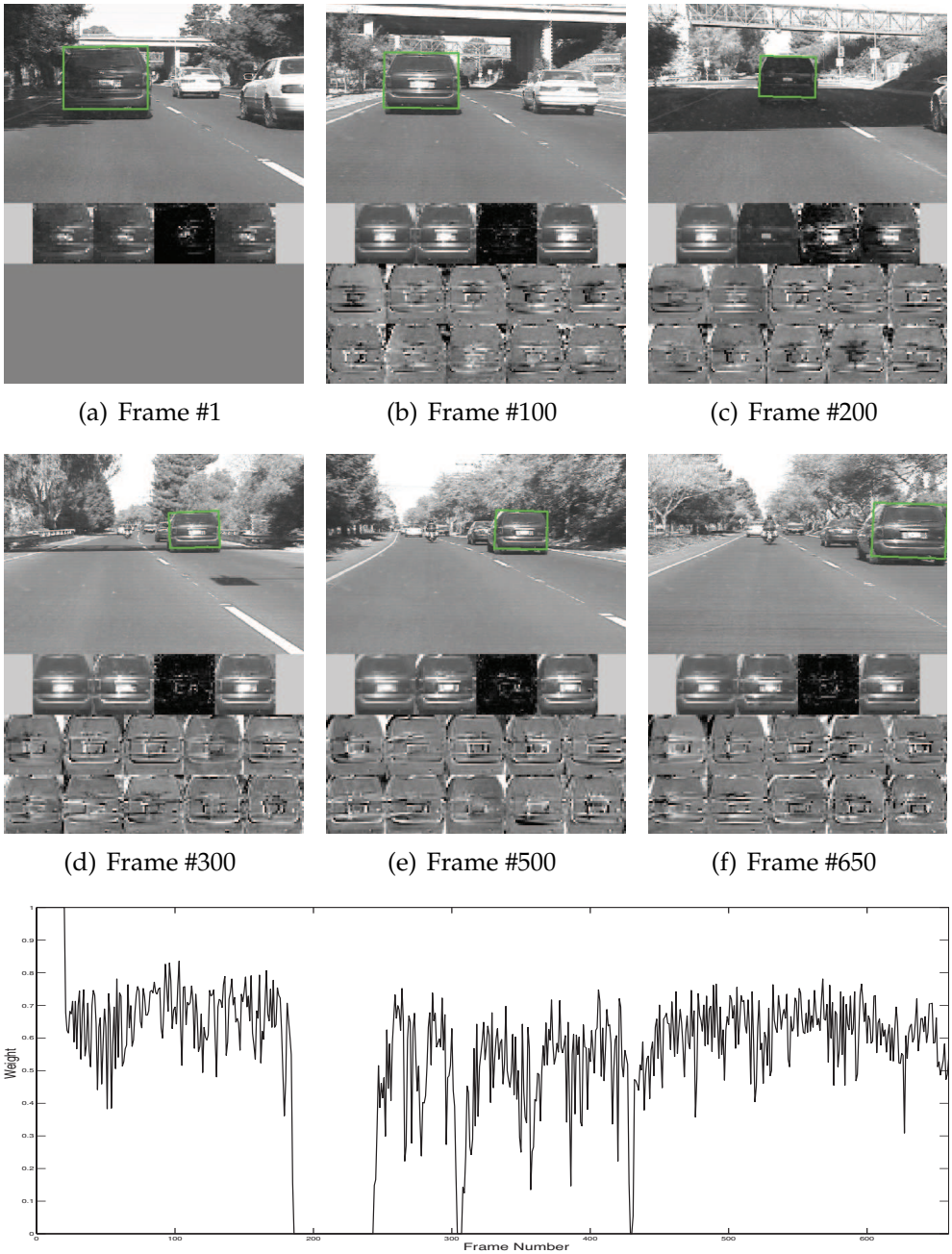


(d) Frame #1064 (e) Frame #1069 (f) Frame #1079



(g) Weights applied to each frame

Figure 10: Example of pedestrian tracking using ITWVTSP/R-iso ($\varepsilon = 0.12$ and ${}^{s}\omega_{\max} = 3.2$) on the sequence S1-T1-C Camera 3 of the PETS2006 Dataset.

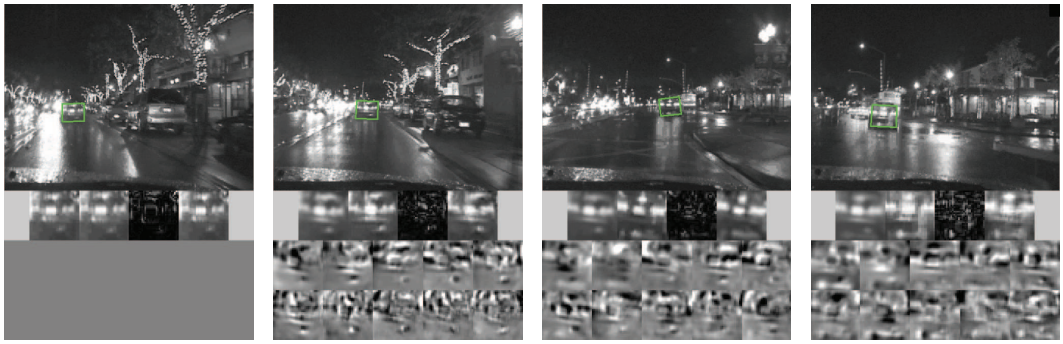


(a) Frame #1 (b) Frame #100 (c) Frame #200

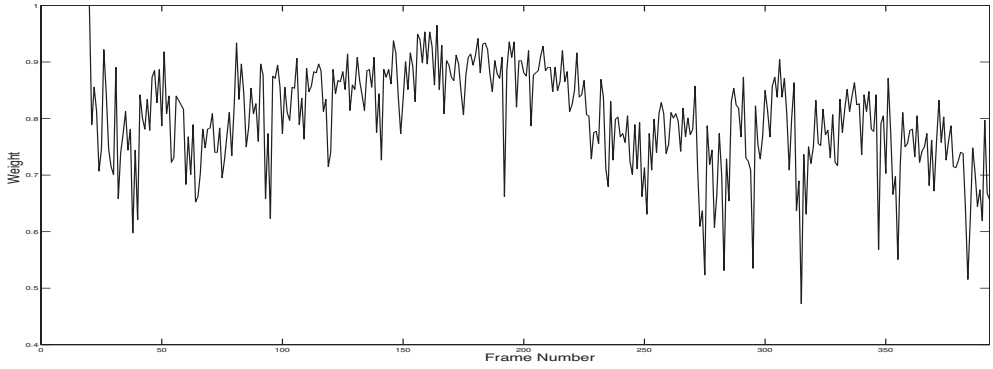
(d) Frame #300 (e) Frame #500 (f) Frame #650

(g) Weights applied to each frame

Figure 11: Example of vehicle tracking using ITWVTSP/R-iso ($\varepsilon = 0.07$ and ${}^{s}\omega_{\max} = 2.0$).



(a) Frame #1 (b) Frame #150 (c) Frame #300 (d) Frame #390



(e) Weights applied to each frame

Figure 12: Example of night vehicle tracking using ITWVTSP/R-iso ($\varepsilon = 0.12$ and ${}^{s}\omega_{\max} = 2.0$).

Several alternatives for the computation of temporal weights and spatial penalty have been introduced, producing a family of VT algorithms. All the alternatives have been 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 have been 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 labelled, the superiority of the proposed approach against state-of-the-art techniques has been clearly shown by means of RMSE, precision and lost track ratio.

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 seems 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 have seen that changes in the appearance of the tracked object, in spatially important regions, can decrease the performance of ITWVTSP compared to ITWVT. This suggest the study of dynamical spatial penalty strategies. Indeed, reconstruction error gives valuable spatial information about changes of the object appearance. This information could be used for adapting dynamically the values of the spatial weights.

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

$$\begin{aligned} S_{Z^{(1,2)}} &= S_{Z^{(1)}} + S_{Z^{(2)}} \\ &+ \frac{\|{}^t\omega^{(1)}\|_1 \|{}^t\omega^{(2)}\|_1}{\|{}^t\omega^{(1)}\|_1 + \|{}^t\omega^{(2)}\|_1} (\mu^{(1)} - \mu^{(2)})(\mu^{(1)} - \mu^{(2)})^\top, \end{aligned} \tag{19}$$

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.

Proof. Note that

$$\mu^{(1,2)} = \frac{\|{}^t\omega^{(1)}\|_1}{\|{}^t\omega^{(1)}\|_1 + \|{}^t\omega^{(2)}\|_1} \mu^{(1)} + \frac{\|{}^t\omega^{(2)}\|_1}{\|{}^t\omega^{(1)}\|_1 + \|{}^t\omega^{(2)}\|_1} \mu^{(2)},$$

and so

$$\mu^{(1)} - \mu^{(1,2)} = \frac{\|{}^t\omega^{(2)}\|_1}{\|{}^t\omega^{(1)}\|_1 + \|{}^t\omega^{(2)}\|_1} (\mu^{(1)} - \mu^{(2)}), \tag{20}$$

and

$$\mu^{(2)} - \mu^{(1,2)} = \frac{\|{}^t\omega^{(1)}\|_1}{\|{}^t\omega^{(1)}\|_1 + \|{}^t\omega^{(2)}\|_1} (\mu^{(2)} - \mu^{(1)}). \tag{21}$$

Then,

$$\begin{aligned} S_{Z^{(1,2)}} &= \sum_{i=1}^{N^{(1)}} \omega_i^{(1)} (z_i^{(1)} - \mu^{(1,2)})(z_i^{(1)} - \mu^{(1,2)})^\top \\ &+ \sum_{i=1}^{N^{(2)}} \omega_i^{(2)} (z_i^{(2)} - \mu^{(1,2)})(z_i^{(2)} - \mu^{(1,2)})^\top \\ &= \sum_{i=1}^{N^{(1)}} \omega_i^{(1)} (z_i^{(1)} - \mu^{(1)} + \mu^{(1)} - \mu^{(1,2)}) \end{aligned}$$

$$
\begin{aligned}
& (z_i^{(1)} - \mu^{(1)} + \mu^{(1)} - \mu^{(1,2)})^\top \\
+ & \sum_{i=1}^{N^{(2)}} \omega_i^{(2)} (z_i^{(2)} - \mu^{(2)} + \mu^{(2)} - \mu^{(1,2)}) \\
& (z_i^{(2)} - \mu^{(2)} + \mu^{(2)} - \mu^{(1,2)})^\top \\
= & S_{Z^{(1)}} + S_{Z^{(2)}} \\
+ & \sum_{i=1}^{N^{(1)}} \omega_i^{(1)} (\mu^{(1)} - \mu^{(1,2)})(\mu^{(1)} - \mu^{(1,2)})^\top \\
+ & \sum_{i=1}^{N^{(2)}} \omega_i^{(2)} (\mu^{(2)} - \mu^{(1,2)})(\mu^{(2)} - \mu^{(1,2)})^\top \qquad (22)
\end{aligned}
$$

Applying Eqs. (20) and (21) on Eq. (22), we obtain

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

□

References

An, J.-H. and Hong, K.-S. (2011). Finger gesture-based mobile user interface using a rear-facing camera, *Consumer Electronics (ICCE), 2011 IEEE International Conference on*, pp. 303–304.

Baseggio, M., Cenedese, A., Merlo, P., Pozzi, M. and Schenato, L. (2010). Distributed perimeter patrolling and tracking for camera networks, *Decision and Control (CDC), 2010 49th IEEE Conference on*, pp. 2093 – 2098.

Birchfield, S. and Rangarajan, S. (2005). Spatiograms versus histograms for region-based tracking, *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, Vol. 2, pp. 1158–1163.

Brand, M. (2006). Fast low-rank modifications of the thin singular value decomposition, *Linear Algebra and Its Applications* **415**(1): 20–30.

Breitenstein, M., Reichlin, F., Leibe, B., Koller-Meier, E. and Gool, L. V. (2009). Robust tracking-by-detection using a detector confidence particle filter, *Computer Vision, 2009 IEEE 12th International Conference on*, pp. 1515–1522.

Comaniciu, D., Ramesh, V. and Meer, P. (2003). Kernel-based object tracking, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **25**(5): 564–577.

Cootes, T., Edwards, G. and Taylor, C. (2001). Active appearance models, *IEEE Trans. Pattern Anal. Mach. Intell.* **23**: 681–685.

Czyz, J., Ristic, B. and Macq, B. (2007). A particle filter for joint detection and tracking of color objects, *Image and Vision Computing* **25**(8): 1271–1281.

Enzweiler, M. and Gavrila, D. (2009). Monocular pedestrian detection: Survey and experiments, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **31**(12): 2179–2195.

Galleguillos, C. and Belongie, S. (2010). Context based object categorization: A critical survey, *Computer Vision and Image Understanding*

114(6): 712–722. Special Issue on Multi-Camera and Multi-Modal Sensor Fusion.

Gerónimo, D., López, A., Sappa, A. and Graf, T. (2010). Survey of pedestrian detection for advanced driver assistance systems, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **32**(7): 1239 –1258.

Grabner, H., Grabner, M. and Bischof, H. (2006). Real-time tracking via online boosting, *British Machine Vision Conference (BMVC)*, Vol. 1, pp. 47–56.

Haj, M. A., Bagdanov, A., Gonzàlez, J. and Roca, F. (2010). Reactive object tracking with a single ptz camera, *Pattern Recognition (ICPR), 2010 20th International Conference on*, pp. 1690–1693.

Hall, P., Marshall, D. and Martin, R. (1998). Incremental Eigenanalysis for Classification, *British Machine Vision Conference*, pp. 286–295.

Huang, D., Yi, Z. and Pu, X. (2009). A new incremental pca algorithm with application to visual learning and recognition, *Neural Processing Letters* **30**: 171–185.

Huang, K., Wang, L., Tan, T. and Maybank, S. (2008). A real-time object detecting and tracking system for outdoor night surveillance, *Pattern Recognition* **41**(1): 432–444.

Hubert, M., Rousseeuw, P. J. and Branden, K. V. (2005). ROBPCA: a new approach to robust principal component analysis, *Technometrics* **47**: 64–79.

Hubert, M., Rousseeuw, P. J. and Verdonck, T. (2009). Robust pca for skewed data and its outlier map, *Computational Statistics & Data Analysis* **53**(6): 2264–2274.

Isard, M. and Blake, A. (1998). CONDENSATION conditional density propagation for visual tracking, *International Journal of Computer Vision* **29**: 5–28.

Iwahori, Y., Enda, N., Fukui, S., Kawanaka, H., Woodham, R. and Adachi, Y. (2008). Efficient tracking with adaboost and particle filter under complicated background, *in* I. Lovrek, R. Howlett and L. Jain (eds),

Knowledge-Based Intelligent Information and Engineering Systems, Vol. 5178 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 887–894.

Jackson, D. A. and Chen, Y. (2004). Robust principal component analysis and outlier detection with ecological data, *Environmetrics* **15**(2): 129–139.

Jepson, A., Fleet, D. and El-Maraghi, T. (2003). Robust online appearance models for visual tracking, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **25**(10): 1296–1311.

Kalal, Z., Matas, J. and Mikolajczyk, K. (2010). P-n learning: Bootstrapping binary classifiers by structural constraints, *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 49 –56.

Kriegel, H., Kriegelöger, P., Schubert, E. and Zimek, A. (2008). A general framework for increasing the robustness of pca-based correlation clustering algorithms, *Proceedings of the 20th international conference on Scientific and Statistical Database Management*, pp. 418–435.

Lee, K.-C., Ho, J., Yang, M.-H. and Kriegman, D. (2005). Visual tracking and recognition using probabilistic appearance manifolds, *Computer Vision and Image Understanding* **99**(3): 303–331.

Levy, A. and Lindenbaum, M. (2000). Sequential karhunen-loeve basis extraction and its application to images, *IEEE Transactions on Image Processing* **9**(8): 1371–1374.

Lucas, B. and Kanade, T. (1981). An iterative image registration technique with an application to stereo vision, *International Joint Conference on Artificial Intelligence*, pp. 674–679.

Maggio, E. and Cavallaro, A. (2010). *Video Tracking: Theory and Practice*, Wiley and Sons.

Marimon, D. and Ebrahimi, T. (2007). Combination of video-based camera trackers using a dynamically adapted particle filter, *Proc. 2nd International Conference on Computer Vision Theory and Applications (VISAPP07)*, pp. 363–370.

Mundy, J. (2006). Object recognition in the geometric era: A retrospective, *in* J. Ponce, M. Hebert, C. Schmid and A. Zisserman (eds), *Toward Category-Level Object Recognition*, Vol. 4170 of *Lecture Notes in Computer Science*, Springer Berlin / Heidelberg, pp. 3–28.

Papadourakis, V. and Argyros, A. (2010). Multiple objects tracking in the presence of long-term occlusions, *Computer Vision and Image Understanding* **114**(7): 835–846.

Park, S. and Aggarwal, J. K. (2004). A hierarchical bayesian network for event recognition of human actions and interactions, *Multimedia Systems* **10**(2): 164–179.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space, *Philosophical Magazine* **2**(6): 559–572.

Peng, N.-S., Yang, J. and Liu, Z. (2005). Mean shift blob tracking with kernel histogram filtering and hypothesis testing, *Pattern Recognition Letters* **26**(5): 605–614.

Polat, E., Yeasin, M. and Sharma, R. (2003). Robust tracking of human body parts for collaborative human computer interaction, *Computer Vision and Image Understanding* **89**(1): 44–69.

Reinartz, P., Lachaise, M., Schmeer, E., Krauss, T. and Runge, H. (2006). Traffic monitoring with serial images from airborne cameras, *ISPRS Journal of Photogrammetry and Remote Sensing* **61**(3-4): 149–158. Theme Issue: Airborne and Spaceborne Traffic Monitoring.

Ross, D. A., Lim, J., Lin, R.-S. and Yang, M.-H. (2008). Incremental learning for robust visual tracking, *International Journal of Computer Vision* **77**(1-3): 125–141.

Santis, A. D. and Iacoviello, D. (2009). Robust real time eye tracking for computer interface for disabled people, *Computer Methods and Programs in Biomedicine* **96**(1): 1–11.

Semertzidis, T., Dimitropoulos, K., Koutsia, A. and Grammalidis, N. (2010). Video sensor network for real-time traffic monitoring and surveillance, *Intelligent Transport Systems, IET* **4**(2): 103 –112.

Skocaj, D., Leonardis, A. and Bischof, H. (2007). Weighted and robust learning of subspace representations, *Pattern Recognition* **40**(5): 1556–1569.

Skočaj, D. and Leonardis, A. (2008). Incremental and robust learning of subspace representations, *Image Vision Comput.* **26**: 27–38.

Stauffer, C. and Grimson, W. (1999). Adaptive background mixture models for real-time tracking, *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on.*, Vol. 2, pp. 637–663.

Tipping, M. E. and Bishop, C. M. (1999). Probabilistic principal component analysis, *Journal of the Royal Statistical Society, Series B* **61**: 611–622.

Tomasi, C. and Kanade, T. (1991). Detection and tracking of point features, *Technical Report CMU-CS-91-132*, International Journal of Computer Vision.

Yang, M.-H., Kriegman, D. and Ahuja, N. (2002). Detecting faces in images: a survey, *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **24**(1): 34–58.

Yilmaz, A., Javed, O. and Shah, M. (2006). Object tracking: A survey, *ACM Comput. Surv.* **38**.

Zhou, S., Chellappa, R. and Moghaddam, B. (2004). Visual tracking and recognition using appearance-adaptive models in particle filters, *Image Processing, IEEE Transactions on* **13**(11): 1491–1506.