Visual Tracking using the Earth Mover's Distance between Gaussian Mixtures and Kalman Filtering

Vasileios Karavasilis, Christophoros Nikou and Aristidis Likas

Department of Computer Science, University of Ioannina,
PO Box 1186, 45110 Ioannina, Greece,
Phone: + (30) 265 100 8802
{vkaravas, cnikou, arly}@cs.uoi.gr

Abstract

In this paper, we demonstrate how the differential Earth Mover's Distance (EMD) may be used for visual tracking in synergy with Gaussian mixtures models (GMM). According to our model, motion between adjacent frames results in variations of the mixing proportions of the Gaussian components representing the object to be tracked. These variations are computed in closed form by minimizing the differential EMD between Gaussian mixtures, yielding a very fast algorithm with high accuracy, without recurring to the EM algorithm in each frame. Moreover, we also propose a framework to handle occlusions, where the prediction for the object's location is forwarded to an adaptive Kalman filter whose parameters are estimated on line by the motion model already observed. Experimental results show significant improvement in tracking performance in the presence of occlusion.

Index Terms

Visual tracking, Gaussian Mixture Models (GMM), Expectation-Maximization (EM) algorithm, Differential Earth Mover's Distance (Differential EMD), Kalman filter.

I. Introduction

One important field in computer vision is tracking. Tracking is the procedure of generating an inference about motion given a sequence of images. Solutions to this problem have a variety of applications, some of them being: surveillance, targeting, recognition from motion, motion-based video compression, teleconferencing, video indexing and traffic monitoring. In tracking problems, it is assumed that the model of the object is known, that is how the object looks like, or its appearance. Based on a set of measurements in image frames the object's position should be estimated. In that context, the Differential Earth Mover's Distance (DEMD) tracking algorithm [40], [41] was recently presented. In this method, the object is represented by a histogram (called a signature) and the distance between signatures in consecutive frames to be minimized is the Earth Mover's Distance [25]. The computational complexity of the EMD prevents a direct implementation in many real time applications. To overcome this drawback, the DEMD algorithm based on sensitivity analysis of the simplex method provides an acceleration compared with its standard counterpart [40], [41].

Motivated by the efficiency of the differential EMD tracking algorithm [40], [41] and the compactness of the representation of probability densities using Gaussian mixture models [5], we propose in this paper to first model the appearance of the target by a Gaussian mixture model trained on a weighted likelihood and then to employ the differential EMD approach for tracking. According to our model, motion between adjacent frames results in variations of the mixing proportions of the Gaussian components representing the object. These variations affect the distance between the mixtures, at the same image location, representing the object in consecutive frames. By these means, the gradient of the EMD, namely the differential EMD [40], [41], between Gaussian mixtures shows the direction of the minimum and consequently the target location.

Moreover, in a second phase of this work, we propose to consider the estimated location of the target as a measurement (observation) of a time-varying Kalman filter in order to address cases presenting occlusions. Hence, the prediction for the object's location is forwarded to a Kalman filter whose state matrix parameters are not constant but they are updated on-line based on recent history of the estimated motion.

The contribution of the presented work is twofold. At first, the proposed approach leads to a significant improvement in terms of execution time with respect to the differential EMD tracking algorithm [40], [41] without compromising the accuracy of the method. At second, based on the motion model already observed, occlusions are successfully handled by modifying on-line the state matrix of a Kalman filter.

The remainder of the paper is organized as follows: A review of tracking algorithms is presented in section II. In section III, the modeling of the object to be tracked by a Gaussian mixture is presented. The tracking algorithm relying on the minimization of the Earth Mover's Distance between Gaussian mixtures is presented in section IV. In section V, the extension of the algorithm in order to address the problem of occlusion is described. Experimental results are shown in section VI which are followed by our conclusion in section VII.

II. Related Work

Tracking algorithms may be classified in two categories [9]. The first category is based on filtering and data association, while the second family of methods relies on target representation and localization. The algorithms based on filtering assume that the moving object has an internal state which may be measured and, by combining the measurements with the model of state evolution, the object's position is estimated. The first method of that category is the Kalman filter [28] which successfully tracks objects even in the case of occlusion if the assumed type of motion is correctly modeled [11]. Another approach in this category are the particle filters [2], [37]. This category also includes Condensation [16] and ICondensation [17] algorithms which are more general than Kalman filters, as they do not assume specific type of densities and, using factored sampling, have the ability to predict an object's location under occlusion as well. Also, in this category, methods based on feature extraction and tracking were also proposed [32]. The object is represented by a set of scale invariant landmarks [18] which are tracked using optical flow [19], [27]. These methods have the drawback that the type of object's movement should be correctly modeled.

On the other hand, tracking algorithms relying on target representation and localization employ a probabilistic model of the object appearance and try to detect this model in consecutive frames of the image sequence. More specifically, color or texture features of the object, masked by an isotropic kernel, are used to create a histogram. Then, the object's position is estimated by minimizing a cost function between the model's histogram and candidate histograms in the next image. A representative method in this category is the mean shift algorithm [9] where the object is supposed to be inside an ellipse and the histogram is constructed from pixel values inside that ellipse. Extensions of the main algorithm are proposed in [34] where the mean shift is combined with particle filters, in [42] where scale invariant features are used and in [36] where various distance measures are associated with the mean shift algorithm. Other approaches using multiple kernels [12] and a Newton style optimization procedure [14] were also proposed. The DEMD tracking algorithm [40], [41] also belongs to the category of methods relying on target representation and localization.

The above methods track only one object at a time. Other works track many objects simultaneously [1], [4], [7], [30] and in these cases occlusions may be detected more efficiently. Moreover, the object to be tracked is usually represented by its color histogram, but this is not always necessary. A Gaussian mixture model (GMM) was used in [33] to represent the object in a joint spatial-color space and in [29] for background subtraction and the contour of the object was tracked in [39], [26]. Moreover, it is well-known that a level set representation also addresses the problem of multiple objects [20], [22], [23], [24], [10]. In any case, combining multiple object representations could make the tracking procedure more robust [15], [21], [31], [35]. Finally, in [6], multiple views of an object are learnt through Principal Component Analysis (PCA) and a Support Vector Machine (SVM) classifier was also used in [3]. A review of tracking methods can be found in [38].

III. Target appearance modeling

In this section we present the basic idea of minimizing the Earth Mover's Distance between Gaussian mixture models for tracking. We describe the GMM as a way of representing an object's appearance and define the EMD as a distance between two GMM.

A. Background on weighted Gaussian Mixture Models

A one dimensional Gaussian distribution has a probability density function given by

$$\mathcal{N}(I_n|\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(I_n-\mu)^2}{2\sigma^2}\right) \tag{1}$$

Where I_n is the intensity of the n^{th} pixel, μ is the mean value and σ^2 is the variance of the distribution.

Let two Gaussian distributions be:

$$f_1(I_n) = \mathcal{N}(I_n|\mu_1,\sigma_1), \qquad f_2(I_n) = \mathcal{N}(I_n|\mu_2,\sigma_2). \tag{2}$$

The Gaussian Mixture Model (GMM) is a convex combination of Gaussian components [5]. A single component is given by (1) and the GMM with m components is expressed by

$$f(I_n|\mu,\sigma,\pi) = \sum_{i=1}^{m}\pi_i\mathcal{N}(I_n|\mu_i,\sigma_i) \tag{3}$$

where $\mu = \{\mu_i\}_{i=1,\ldots,m}$, $\sigma = \{\sigma_i\}_{i=1,\ldots,m}$ and $\pi = \{\pi_i\}_{i=1,\ldots,m}$, are the model parameters. The parameters π_i represent the importance of each component and satisfy the constraints $\sum_{i=1}^{m}\pi_i = 1$ and $\pi_i \geq 0$, $\forall i = 1,\ldots,m$.

We assume that we have grayscale images, and each object may be described by the intensities of its pixels. An object is represented by an ellipsoidal region, and the object's pixels are those lying inside that region. The usual way to represent an object is by histograms of m_h bins. This approach has the disadvantage that the number of the bins must be specified *a priori*. However, it is a very common and efficient way of modeling the object to be tracked in the majority of the state of the art trackers [9].

In this work we propose the representation of an object using a GMM. The parameters of the GMM are estimated by clustering the density values of object's pixels using the EM algorithm [5]. An advantage of the GMM representation is that the number of components m is significantly smaller than the number of distinct intensities.

Every object may be represented by an ellipsoidal region with finite precision. As an effect, inside the ellipse, there will be regions not belonging to the object. Usually, these regions exist at the edges of the ellipse. To eliminate the influence of regions not belonging to the object, the ellipse is weighed by a kernel as will be explained bellow.

At first, we assume that the center of the ellipse is in the spatial location $(0, 0)$. Then, the ellipse is normalized to a unit circle by dividing each pixel coordinates by h_x and h_y, which are the sizes of the ellipse in the horizontal and vertical directions respectively. Let the normalized pixel locations be (x_n, y_n). An isotropic kernel, with profile $k(x)$, is applied to pixels inside the unit circle to attribute corresponding weights at every pixel. The weight for a pixel indexed by n is defined by

$$w_n = \frac{k(x_n^2 + y_n^2)}{\sum_{i=1}^{N} k(x_i^2 + y_i^2)}. \tag{4}$$

Notice that $x_n^2 + y_n^2 \leq 1$ because the point (x_n, y_n) is inside unit sphere and $\sum_{n=1}^{N} w_n = 1$. The kernel profile $k(x)$ is a convex monotonic decreasing function such that $k : [0, \infty) \to \Re$ and g is the negative derivative of the kernel function, $g(x) = -k'(x)$. We use a kernel with Epanechnikov profile [9]

$$k(x) = \begin{cases} \frac{1}{2}(1 - x) & if \quad x \leq 1 \\ 0 & otherwise \end{cases} \tag{5}$$

Let $\mathbf{I} = \{I_n\}_{n=1,\ldots,N}$ be the intensity values of the pixels inside the unit circle as defined above and let $\mathbf{W} = \{w_n\}_{n=1,\ldots,N}$ be the corresponding normalized weights of each sample. The weighted likelihood of the model is expressed by

$$\begin{aligned} L(\mathbf{I}, \mathbf{W}|\mu, \sigma, \pi) &= \sum_{n=1}^{N} w_n \log\left(f(I_n|\mu, \sigma, \pi)\right) \\ &= \sum_{n=1}^{N} w_n \log\left(\sum_{i=1}^{m} \pi_i \mathcal{N}(I_n|\mu_i, \sigma_i)\right) \end{aligned} \tag{6}$$

and the update equations of the Expectation - Maximization (EM) algorithm that maximize this likelihood are:

- Expectation step: Compute responsibilities

$$\gamma_{ni} = \frac{\pi_i \mathcal{N}(I_n|\mu_i, \sigma_i)}{\sum_{j=1}^{m} \pi_j \mathcal{N}(I_n|\mu_j, \sigma_j)}. \tag{7}$$

- Maximization step: Estimate parameters

$$\hat{\pi}_i = \sum_{n=1}^{N} w_n \gamma_{ni}, \tag{8}$$

$$\hat{\mu}_i = \frac{\sum_{n=1}^{N} w_n \gamma_{ni} I_n}{\sum_{n=1}^{N} w_n \gamma_{ni}}, \tag{9}$$

$$\hat{\sigma}_i^2 = \frac{\sum_{n=1}^{N} w_n \gamma_{ni} (I_n - \hat{\mu}_i)^2}{\sum_{n=1}^{N} w_n \gamma_{ni}}. \tag{10}$$

The above iterations are repeated until convergence of the likelihood. In our method the EM algorithm is applied at the initialization step and when significant changes are observed, in order to infer the GMM parameters that are to be tracked in the following frames.

B. Earth Mover's Distance between Gaussian Mixture Model

Having computed the parameters $\mu = \{\mu_i\}_{i=1,\ldots,m}$, $\sigma = \{\sigma_i\}_{i=1,\ldots,m}$, $\pi^M = \{\pi_i^M\}_{i=1,\ldots,m}$ of the object's model, in the next frame we assume that the new center of the ellipse comprising the target is located at the normalized coordinates $\mathbf{y}$ of the next frame. In the above notation, the exponent M in π^M represents the object's model. We assume that the colors of the object and the background do not change abruptly, so the target GMM candidates have the same mean and variance as their counterpart in the initial frame and the only difference is the importance (mixing proportion) of each component. The model for the background may change between frames but it should have limited overlap with the model of the object. The only problem is when pixels from the background have the same intensity with pixels belonging to the object (camouflage). In case the mixing proportions of the GMM do not change smoothly between frames, this is an indication that important illumination changes occur. Therefore, the GMM needs to be trained again to take into account the new illumination conditions.

An illustrative example is presented in figure 1, where the number of components $m = 3$, highlights the change in each component's importance. The value for m is appropriate for this example. This parameter actually depends on the colors of the object (e.g. it can be determined by the number of colors belonging to the object and those belonging to the background). Statistical criteria may also be employed in order to estimate more accurately the number of components [13]. In the images at the left, the ellipse remains at the same spatial location, while the racket is moving downwards. In the right figures, the horizontal axis represents the gray levels and the vertical axis represents the probability of each gray level (3). Each GMM has three components. As the racket is moving outside of the ellipse, the mixing proportions associated with the object get smaller while the mixing proportion representing the background is increasing.

Therefore, the GMM parameters for the candidate object in the next frame are (described by an exponent C) are $\mu = \{\mu_i\}_{i=1,\ldots,m}$, $\sigma = \{\sigma_i\}_{i=1,\ldots,m}$, $\pi^C(\mathbf{y}) = \{\pi_i^C(\mathbf{y})\}_{i=1,\ldots,m}$, where the mixing proportions depend on the location $\mathbf{y}$. This means that the centers μ_i and the variances σ_i^2 remain unchanged through time. Also, by assuming that $\pi_i^C(\mathbf{y})$ do not change dramatically through time, equations (7) and (8) of the EM algorithm may be used to estimate the proportions $\pi_i^C(\mathbf{y})$. We must point out that the EM is used only in the initial image. In all other images, only computation of the

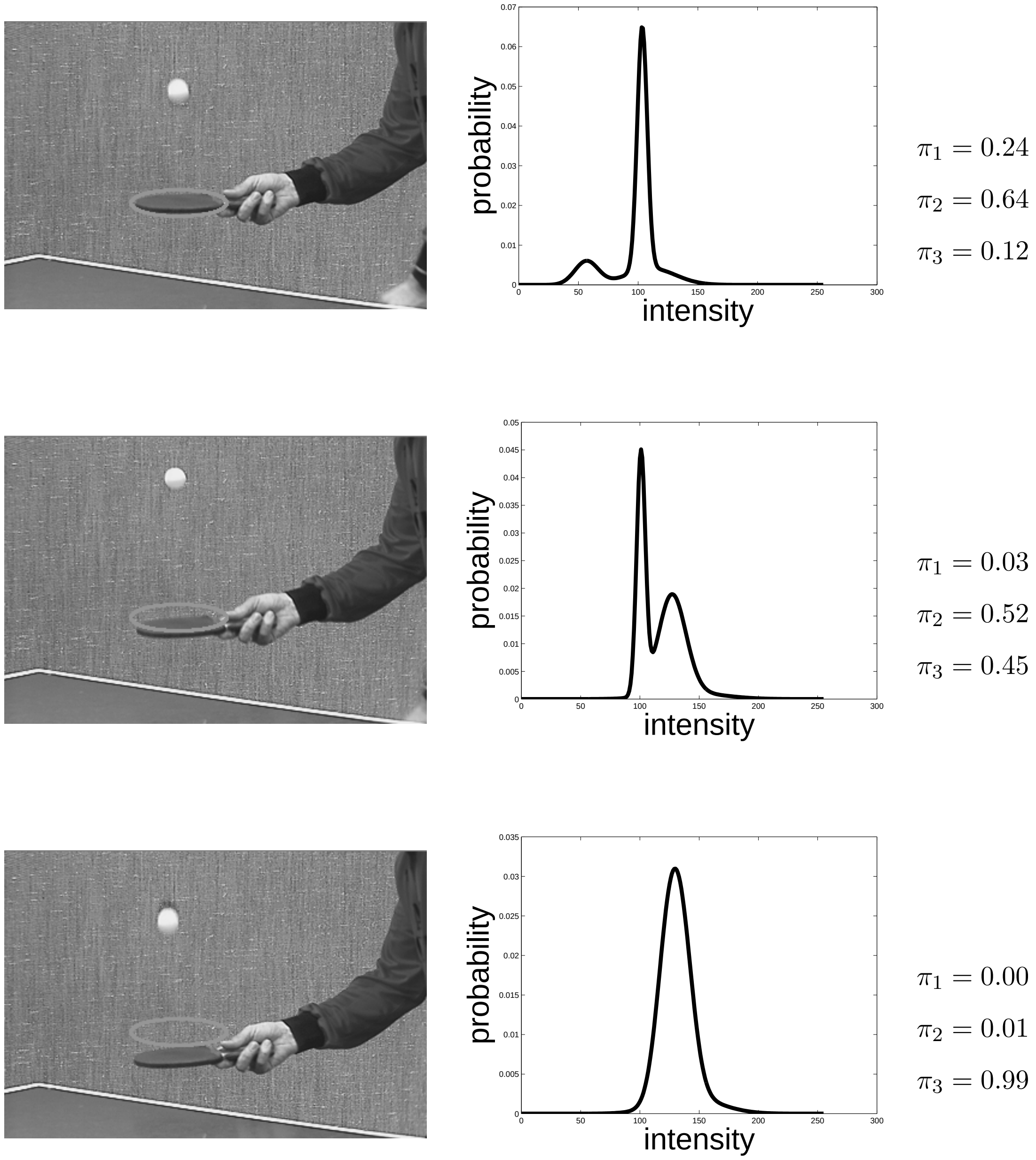


Fig. 1. Variations of the GMM parameters during tracking. As the racket moves, the component that corresponds to the background (π_3) increases its proportion in the GMM due to the fact that more pixels belonging to background are inside the ellipse. On the other hand, components corresponding to the object (π_1 and π_2) reduce their responsibilities γ_{ni} in the model because pixels belonging to the object get out of the ellipse. Nevertheless, the means and variances of the model components remain unchanged because the object and background colors change smoothly.

proportion $\pi^C(\mathbf{y})$ is made, as means μ and variances σ remain unchanged (due to the fact that the color and the luminance of the object remain unchanged). By substituting $\pi_i \leftarrow \pi_i^M$ and $\hat{\pi}_i \leftarrow \pi_i^C(\mathbf{y})$

in (7) and (8) respectively, the mixing proportions for the candidate object is:

$$\pi_i^C(\mathbf{y}) = \frac{1}{N} \sum_{n=1}^{N} w_n^C(\mathbf{y}) \frac{\pi_i^M \mathcal{N}(I_n^C(\mathbf{y})|\mu_i, \sigma_i)}{\sum_{j=1}^m \pi_j^M \mathcal{N}(I_n^C(\mathbf{y})|\mu_j, \sigma_j)} \tag{11}$$

where $I_n^C(\mathbf{y})$ is the image intensity of the n^{th} pixel of the candidate object at location $\mathbf{y}$, $w_i^C(\mathbf{y})$ are the normalized pixel weights inside the unit circle in the next image and N is the number of the pixels.

As the means μ_i and the variances σ_i^2 of the GMM in the initial and the current frames are the same, the parameters for the first GMM are defined by $\mu = \{\mu_i\}_{i=1,\dots,m}$, $\sigma = \{\sigma_i\}_{i=1,\dots,m}$, $\pi^M = \{\pi_i^M\}_{i=1,\dots,m}$ and for the second one are $\mu = \{\mu_i\}_{i=1,\dots,m}$, $\sigma = \{\sigma_i\}_{i=1,\dots,m}$, $\pi^C(\mathbf{y}) = \{\pi_i^C(\mathbf{y})\}_{i=1,\dots,m}$. We define the Earth Mover's Distance (EMD) between two GMM as [25]:

$$EMD(\mathbf{y}) = \min_{f_{uv}} \left[\sum_{u=1}^{m} \sum_{v=1}^{m} f_{uv}(\mathbf{y}) d_{uv} \right] \tag{12}$$

subject to

$$\begin{aligned}
&\textstyle\sum_{u=1}^m f_{uv}(\mathbf{y}) = \pi_v^C(\mathbf{y}), && 1 \le v \le m \\
&\textstyle\sum_{v=1}^m f_{uv}(\mathbf{y}) = \pi_u^M, && 1 \le u \le m \\
&\textstyle\sum_{u=1}^m \sum_{v=1}^m f_{uv}(\mathbf{y}) = 1 && \\
&f_{uv}(\mathbf{y}) \ge 0, && 1 \le u \le m, 1 \le v \le m
\end{aligned} \tag{13}$$

where d_{uv} is the symmetric Kullback-Leibler distance given by

$$d_{u,v} = \frac{1}{2} \left[\frac{\sigma_u^2}{\sigma_v^2} + \frac{\sigma_v^2}{\sigma_u^2} + (\mu_u - \mu_v)^2 \left(\frac{1}{\sigma_u^2} + \frac{1}{\sigma_v^2} \right) - 2 \right] = D_{KL}(f_u|f_v) + D_{KL}(f_v|f_u), \tag{14}$$

where $D_{KL}(f_1|f_2)$ is the Kullback-Leibler divergence [5] between f_1 and f_2 is defined as

$$D_{KL}(f_1|f_2) = \frac{1}{2} \left[\log\left(\frac{\sigma_2^2}{\sigma_1^2} \right) + \frac{\sigma_1^2}{\sigma_2^2} + \frac{(\mu_2 - \mu_1)^2}{\sigma_2^2} - 1 \right]. \tag{15}$$

The product $f_{uv}(\mathbf{y}) d_{uv}$ represents the work needed to transfer a quantity of $f_{uv}(\mathbf{y})$ amount of solid to a distance d_{uv}. These transfers must be performed in such a way that the total work is minimum. Hence, $EMD(\mathbf{y})$ represents the work which must be produced to fill the holes of the second GMM using earth of the first GMM. We must notice that this fill is always possible because $\sum_{u=1}^m \pi_u^M = 1$ and $\sum_{v=1}^m \pi_v^C(\mathbf{y}) = 1$. In other words, the amount of earth in the hills is exactly equal to the amount needed by the holes to be fulfilled.

IV. Target tracking

To locate the target, we must find the ellipse with center located at $\hat{\mathbf{y}}$ which is most similar to the ellipse of the model. In other words, a local minimum of $EMD(\mathbf{y})$ must be found:

$$\hat{\mathbf{y}} = \underset{\mathbf{y}}{\operatorname{argmin}} [EMD(\mathbf{y})] \tag{16}$$

The computation of the EMD between the GMM of the target model and the GMM of the target candidate is computationally expensive if it is repeated for every possible location $\mathbf{y}$ in the target frame. In order to accelerate the procedure, following the principles proposed in [40], [41], we initialize the center $\mathbf{y}$ at the old center estimated in the previous frame, we calculate the gradient of the $EMD(\mathbf{y})$ with respect to $\mathbf{y}$ and use it to determine a new location. The same step is repeated until the value of $EMD(\mathbf{y})$ in the new location increases.

To solve the optimization problem (16), we have to calculate the derivative $\nabla_{\mathbf{y}} EMD(\mathbf{y})$ and choose the neighbor pixel in the direction of the derivative. Using the chain rule [40], [41] yields

$$\nabla_{\mathbf{y}} EMD(\mathbf{y}) = \sum_{v=1}^{m} \frac{\partial EMD(\mathbf{y})}{\partial \pi_v^C(\mathbf{y})} \nabla_{\mathbf{y}} \pi_v^C(\mathbf{y}) \tag{17}$$

The calculation of $\nabla_{\mathbf{y}} EMD(\mathbf{y})$ is described in [40], [41]. Here we summarize the key steps. The difference with [40], [41] is the usage of GMM instead of color signatures, which yields to a different formula for the computation of $\nabla_{\mathbf{y}} \pi_v^C(\mathbf{y})$ and consecutively for the computation of $\nabla_{\mathbf{y}} EMD(\mathbf{y})$. The formula for $\frac{\partial EMD(\mathbf{y})}{\partial \pi_v^C(\mathbf{y})}$ is the same as in [40], [41]. At first, equation (12) and the constraints in (13) are transformed to matrix-vector form. There are $m \times m$ variables $f_{uv}(\mathbf{y})$ and $m \times m$ constants d_{uv} stacked in vectors $\mathbf{f}(\mathbf{y})$ and $\mathbf{d}$ both of size $m^2 \times 1$. Taking together the first three constraints in (13), a matrix ($\mathbf{S}$ of size $m^2 + 1) \times m^2$ is created whose elements are 0 or 1. We also define the vector $\mathbf{b}(\mathbf{y}) = [(\pi^C(\mathbf{y}))^T, (\pi^M)^T, 1]^T$. Using these notations, equation (12) may be written as

$$EMD(\mathbf{y}) = \min_{\mathbf{f}} \mathbf{d}^T \mathbf{f}(\mathbf{y}) \tag{18}$$

and the constraints in (13) now become

$$\begin{aligned} \mathbf{S}\mathbf{f}(\mathbf{y}) &= \mathbf{b}(\mathbf{y}) \\ \mathbf{f}(\mathbf{y}) &\geq \mathbf{0} \end{aligned} \tag{19}$$

The above linear programming problem is solved by the simplex method [8]. Since the matrix $\mathbf{S}$ has rank $2m - 1$, there are $2m - 1$ basic variables, which will be denoted by $\mathbf{f}_B(\mathbf{y})$. Also there are $m^2 - 2m + 1$ non basic variables, which will be denoted by $\mathbf{f}_{NB}(\mathbf{y})$. Similarly, we denote by $\mathbf{d}_B$ and $\mathbf{d}_{NB}$ the elements of vector $\mathbf{d} = [\mathbf{d}_B \mathbf{d}_{NB}]^T$. Finally, $\mathbf{S}_B$ and $\mathbf{S}_{NB}$ are the columns of matrix $\mathbf{S}$ corresponding to the basic and non basic variables $\mathbf{f}_B(\mathbf{y})$ and $\mathbf{f}_{NB}(\mathbf{y})$ respectively. Equation (19) can now be written as

$$\begin{bmatrix} \mathbf{S}_B & \mathbf{S}_{NB} \end{bmatrix} \begin{bmatrix} \mathbf{f}_B(\mathbf{y}) \\ \mathbf{f}_{NB}(\mathbf{y}) \end{bmatrix} = \mathbf{b}(\mathbf{y}) \tag{20}$$

By performing sensitivity analysis the derivatives $\frac{\partial EMD(\mathbf{y})}{\partial \pi_v^C(\mathbf{y})}$ are computed as [40], [41]:

$$\frac{\partial EMD(\mathbf{y})}{\partial \pi_v^C(\mathbf{y})} = k_v - \sum_{\substack{j=1 \\ j \neq v}}^{m} k_j \frac{b_j}{\sum_{\substack{l=1 \\ l \neq v}}^{m} b_l} \tag{21}$$

where $k_v = \sum_{l=1}^{2m-1} (\mathbf{d}_B)_l (\mathbf{S}_B^{-1})_{lv}$.

To calculate $\nabla_{\mathbf{y}} \pi_v^C(\mathbf{y})$, which is the different part of our method with respect to the original DEMD paper [41], the derivative of (11) with respect to $\mathbf{y}$ must be computed. After some algebraic manipulation this leads to

$$\nabla_{\mathbf{y}} \pi_v^C(\mathbf{y}) = \frac{1}{N} \sum_{n=1}^{N} \left[B_n C_{n,v} + w_n A_{n,v} I_n^{\prime C}(\mathbf{y}) \right], \tag{22}$$

where

$$w_n = \frac{k(\|x_n - \mathbf{y}\|^2)}{\sum_{i=1}^{m} k(\|x_i - \mathbf{y}\|^2)}, \tag{23}$$

$$A_{n,v} = \sum_{i=1}^{m} \left[C_{n,v}\; C_{i,v} \left[\frac{I_n^C(\mathbf{y}) - \mu_v}{\sigma_v^2} - \frac{I_n^C(\mathbf{y}) - \mu_i}{\sigma_i^2} \right] \right], \tag{24}$$

$$B_n = \frac{2g(\|x_n - \mathbf{y}\|^2)(x_n - \mathbf{y}) \sum_{i=1}^{N} k(\|x_i - \mathbf{y}\|^2) - 2k(\|x_n - \mathbf{y}\|^2) \sum_{i=1}^{N} g(\|x_n - \mathbf{y}\|^2)(x_i - \mathbf{y})}{\left[\sum_{i=1}^{N} k(\|x_i - \mathbf{y}\|^2) \right]^2}, \tag{25}$$

$$C_{n,v} = \frac{\pi_v^M N(I_n^C(\mathbf{y}) | \mu_v, \sigma_v)}{\sum_{j=1}^{m} \pi_j^M N(I_n^C(\mathbf{y}) | \mu_j, \sigma_j)}. \tag{26}$$

In the above equations, $I_n^{\prime C}(\mathbf{y})$ is the spatial derivative of the intensity of the n^{th} pixel of the candidate object at location $\mathbf{y}$.

Substituting (21) and (22) in (17) yields the gradient of the EMD energy in closed form:

$$\nabla_{\mathbf{y}} EMD(\mathbf{y}) = \frac{1}{N} \sum_{n=1}^{N} \left[B_n \sum_{v=1}^{m} \left[\frac{\partial EMD(\mathbf{y})}{\partial \pi_v^C(\mathbf{y})} C_{n,v} \right] + w_n I_n^{\prime C} \sum_{v=1}^{m} \left[\frac{\partial EMD(\mathbf{y})}{\partial \pi_v^C(\mathbf{y})} A_{n,v} \right] \right] \tag{27}$$

The overall tracking algorithm is summarized in algorithm 1. We call this algorithm Mixture-based DEMD (MDEMD). After the computation of the derivative, one of the eight neighbor pixels is chosen. This pixel is the one that its center is most closer to the line that is defined by the gradient.

Algorithm 1 Differential EMD with GMM (MDEMD)

```
    Input: The center $\mathbf{y}^{i-1}$ of the object in the previous frame $i-1$ and the GMM parameters of the object to be tracked.
    Output: The center $\mathbf{y}^i$ of the object in the current frame $i$.
1   Initialization: Set $\mathbf{y}_0 = \mathbf{y}^{i-1}$ and evaluate $EMD(\mathbf{y}_0)$ using (12).
2   Compute $\nabla_{\mathbf{y}} EMD(\mathbf{y}_0)$ using (27).
3   Choose one of the $8$ neighbors of $\mathbf{y}_0$ in the direction of the gradient $\nabla_{\mathbf{y}} EMD(\mathbf{y}_0)$. Let $\mathbf{y}_1$ be the coordinates of this pixel. Evaluate $EMD(\mathbf{y}_1)$ using (12).
4   If $EMD(\mathbf{y}_1) < EMD(\mathbf{y}_0)$ set $\mathbf{y}_0 \leftarrow \mathbf{y}_1$ and goto step 2.
    Else return $\mathbf{y}^i \leftarrow \mathbf{y}_0$.
```

In order to handle target scaling changes, the main idea is to try different sizes for the ellipse and select the one with the minimum EMD. An extension to the notation must be used to introduce the time variable. At time t, the ellipse has axes h_x^t and h_y^t. The canonical coordinates x_n (used by algorithm 1) of the pixels inside the ellipse at time t, are computed by taking into account h_x^t and h_y^t. The current ellipse, representing the object, is obtained using algorithm 1 (MDEMD). Then, two new GMMs are constructed. The first GMM is trained using the pixels of a smaller ellipse (same center, smaller axes with respect to the current ellipse), while the other is trained using the pixels of a bigger ellipse (same center, bigger axeswith respect to the current ellipse). If both of the new GMMs have greater EMD with the initial GMM compared to the EMD of the current ellipse with the initial GMM then the procedure stops. Otherwise the ellipse with the smaller EMD is selected and this procedure is repeated. This procedure is summarized in algorithm 2.

Algorithm 2 Scale adaptation on MDEMD

```
    Input: The center $\mathbf{y}^{i-1}$ and the axes size $h_x^{i-1}$ and $h_y^{i-1}$ of the object in the previous frame $i-1$, and the GMM parameters of the object to be tracked.
    Output: The center $\mathbf{y}^i$ and the axes size $h_x^i$ and $h_y^i$ of the object in the current frame $i$.
1   Initialization: Set $\mathbf{y}_0 = \mathbf{y}^{i-1}$, $h_x = h_x^{i-1}$ and $h_y = h_y^{i-1}$.
2   Call MDEMD with input $\mathbf{y}_0$ and axes size $h_x$ and $h_y$. Store the center return by MDEMD to $\mathbf{y}^i$.
3   Compute $e = EMD(\mathbf{y}^i)$ using (27), using size axes $h_x$ and $h_y$.
4   Compute $e^+ = EMD(\mathbf{y}^i)$ using (27), using size axes $1.1h_x$ and $1.1h_y$.
5   Compute $e^- = EMD(\mathbf{y}^i)$ using (27), using size axes $0.9h_x$ and $0.9h_y$.
6   if $e^+ < e^-$ and $e^+ < e$, set $\mathbf{y}_0 = \mathbf{y}^i$ and axes size $h_x = 1.1h_x$ and $h_y = 1.1h_x$. Go to step 2.
7   if $e^- < e^+$ and $e^- < e$, set $\mathbf{y}_0 = \mathbf{y}^i$ and axes size $h_x = 0.9h_x$ and $h_y = 0.9h_x$. Go to step 2.
8   return $\mathbf{y}^i$ and the axes size $h_x^i = h_x$ and $h_y^i = h_y$
```

V. Robustness to Occlusions

In this section we combine the differential MEMD algorithm with a Kalman filter to handle occlusions.

A. Background on Kalman filter

In general, we assume that there is a linear process governed by an unknown inner state producing a set of measurements. More specifically, there is a discrete time system and its state at time n is given by vector $\mathbf{x}_n$. The state in the next time step $n+1$ is given by

$$\mathbf{x}_{n+1} = \mathbf{F}_n \mathbf{x}_n + \mathbf{w}_n \tag{28}$$

where $\mathbf{F}_n$ is the transition matrix from state $\mathbf{x}_n$ to $\mathbf{x}_{n+1}$ and $\mathbf{w}_n$ is white Gaussian noise with zero mean and covariance matrix $\mathbf{Q}_n$.

The measurement vector $\mathbf{z}_n$ is given by

$$\mathbf{z}_n = \mathbf{H}_n \mathbf{x}_n + \mathbf{v}_n \tag{29}$$

where $\mathbf{H}_n$ is the measurement matrix and $\mathbf{v}_n$ is white Gaussian noise with zero mean and covariance matrix $\mathbf{R}_n$. In equation (29), the measurement $\mathbf{z}_n$ depends only on the current state $\mathbf{x}_n$ and the noise vector $\mathbf{v}_n$ is independent of the noise $\mathbf{w}_n$.

The Kalman filter approach computes the minimum mean-square error estimate of the state $\mathbf{x}_k$ given the measurements $\mathbf{z}_1, \ldots, \mathbf{z}_k$. The solution is obtained using a recursive procedure [28].

B. Differential EMD with GMM and Kalman filter

The main idea behind the combined approach is to find the position of the object with algorithm 1 (measurement) and forward it to Kalman filter to obtain the current position of the object (estimation). The transition matrix $\mathbf{F}_n$ is not known in the beginning and is estimated by the algorithm.

We assume that the object is described by its center coordinates (x, y) and the axes (h_x, h_y) of the ellipse around it and that the size of the ellipse does not change through time. The state vector $\mathbf{x}_n = [x_n, y_n, 1]^T$ is the position of the center in the image in homogenous coordinates (x_n and y_n are the horizontal and vertical coordinate respectively) and its position varies over time as described in equation (28). The matrix $\mathbf{F}_n$ is defined as:

$$\mathbf{F}_n = \begin{bmatrix} 1 & 0 & dx_n \\ 0 & 1 & dy_n \\ 0 & 0 & 1 \end{bmatrix}$$

where dx_n dy_n are the horizontal and vertical translations of the object's center. Parameters dx_n dy_n are not constant in time, but they are computed dynamically as it will be explained below. The noise vector $\mathbf{w}_n = [w_{n_x}, w_{n_y}, 1]^T$ has covariance matrix $\mathbf{Q}$.

We employ algorithm 1 to obtain the measurement vector $\mathbf{z}_n = [x'_n, y'_n]^T$ where x'_n and y'_n are the horizontal and vertical coordinates of the ellipse center. In general, these measurements differ from the state variables x_n and y_n of vector $\mathbf{x}_n$ due to the presence of noise $\mathbf{v}_n$. The relation between measurement $\mathbf{z}_n$ and state $\mathbf{x}_n$ is given by (29), where

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

and the measurement noise $\mathbf{v}_n = [v_{n_x}, v_{n_y}]^T$ has covariance matrix $\mathbf{R}$.

The only problem that remains to be solved is the automatic evaluation of dx_n and dy_n. Using algorithm 1 we obtain:

- the measurement $\mathbf{z}_n$,
- the distance between the mixture components of the model of the target and the target candidate.

The main idea is to use the computed distance to determine if the object was found or not. This provides a quality measure of the current estimate of the object. If the distance is small, then we have a good chance that the object's center is near the predicted center. If this distance is large, then, the target is lost. This distance is expressed as a normalized coefficient:

$$a(\mathbf{y}) = \exp(-cEMD(\mathbf{y})) \tag{30}$$

where $EMD(\mathbf{y})$ is given by (12) and it is the EMD distance between the source and target GMM at position $\mathbf{y}$ and c is a constant. The role of parameter c in eq. (30) is important as it adjusts the influence of the observation in cases where $EMD(\mathbf{y})$ has a relatively large value. In the experiments performed in this work, a value for c around 10 is a good compromise because it leads to a very small value of $a(\mathbf{y})$ when the object is not practically observed, which is desirable. The value of $a(\mathbf{y})$ is an estimation of how confident we are that the object is found. If we are not sure the object is correctly located, then we follow the previous movement of the object, assuming that occlusion took place. On the other hand, when we are convinced that the object is inside the ellipse, we update our knowledge about the object's movement. Relying on the value of a in (30), parameter $\mathbf{d}_n = [dx_n, dy_n]^T$ is automatically updated by:

$$\mathbf{d}_{n+1} = (1 - a(\hat{\mathbf{x}}_n))\mathbf{d}_n + a(\hat{\mathbf{x}}_n)(\hat{\mathbf{x}}_n - \hat{\mathbf{x}}_{n-1}) \tag{31}$$

where $\hat{\mathbf{x}}_n$ is the vector containing the estimated values of the horizontal and vertical coordinates of the ellipse center at time n. In view of (31), the estimate $\hat{\mathbf{x}}_n$ contributes to the updates of the displacement $\mathbf{d}_n$ only when the current estimate resembles the source object model, that is when $a(\hat{\mathbf{x}}_n) \to 1$. On the other hand when $a(\hat{\mathbf{x}}_n) \to 0$, the displacements included in the state matrix $\mathbf{F}_n$ remain nearly unchanged, as they were in step $n-1$, assuming that the object is occluded. This process has the advantage that the matrix $\mathbf{F}_n$ incorporating information on the object movement can be updated by the tracking algorithm.

Algorithm 3 summarizes the differential MEMD tracking algorithm with Kalman filtering. Note that step 4 uses a tracking algorithm to estimate the position of the object. This algorithm can be also mean shift or DEMD.

Algorithm 3 Differential EMD with GMM and Kalman filter

```
1   Initialization: $\hat{\mathbf{x}}_0$ = initial object location:
        $\mathbf{P}_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{Q} = \begin{bmatrix} h_x & 0 & 0 \\ 0 & h_y & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} h_x & 0 & 0 \\ 0 & h_y & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{F}_0 = \mathbf{I}_{3\times 3}.$
2   Compute initial $GMM_0$ in the first frame as described in section III-A.
3   Prediction:
        $\hat{\mathbf{x}}_n^- = \mathbf{F}_n \hat{\mathbf{x}}_{n-1},$
        $\mathbf{P}_n^- = \mathbf{F}_n \mathbf{P}_{n-1} \mathbf{F}_n^T + \mathbf{Q},$
        $\mathbf{G}_n = \mathbf{P}_n^- \mathbf{H}_n^T [\mathbf{H}_n \mathbf{P}_n^- \mathbf{H}_n^T + \mathbf{R}]^{-1}.$
4   Measurement: Compute the new center ($\mathbf{z}_n$), the new $GMM$ and the distance between
    $GMM$ and $GMM_0$ using MDEMD (Algorithm 1).
5   Estimation:
        $\hat{\mathbf{x}}_n = \hat{\mathbf{x}}_n^- + \mathbf{G}_n(\mathbf{z}_n - \mathbf{H}_n \hat{\mathbf{x}}_n^-),$
        $\mathbf{P}_n = (\mathbf{I} - \mathbf{G}_n \mathbf{H}_n)\mathbf{P}_n^-.$
    The output $\hat{\mathbf{x}}_n$ is the object's new location.
6   Update the elements of $\mathbf{F}_n$ using (31).
    Goto the Prediction step for the next iteration.
```

Scale changes could also be handled directly in the Kalman filter. In that case, the state matrix $\mathbf{F}_n$ and the observation matrix $\mathbf{H}$ should be changed accordingly to take into account the scale parameter. The state vector would be $\mathbf{x}_n = [x_n, y_n, h_n, 1]^T$, where h_n is the scale parameter at time n. The state evolution matrix would then be:

$$\mathbf{F}_n = \begin{bmatrix} 1 & 0 & 0 & dx_n \\ 0 & 1 & 0 & dy_n \\ 0 & 0 & 1 & dh_n \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where dh_n is the scale change between time instants $n-1$ and n. The observation matrix would be accordingly:

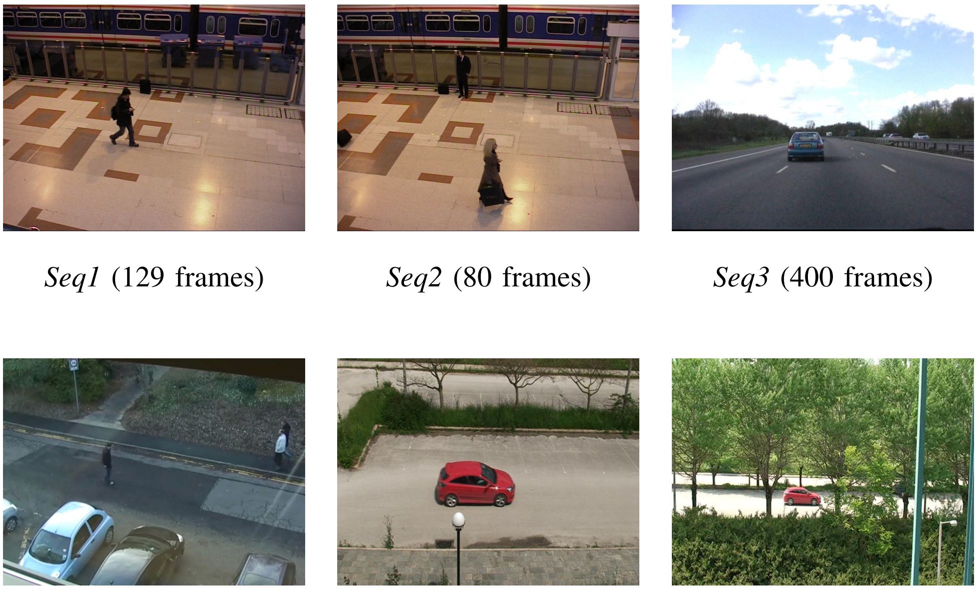
$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

The measurement vector would also contain the scale parameter $\mathbf{z}_n = [x_n, y_n, h_n]^T$, as it would also be the case for the parameter vector $\mathbf{d}_n = [x_n, y_n, h_n]^T$ in eq. (31) which is still valid. Finally, in step 4 of Algorithm 3, the new center and scale would be computed using Algorithm 2 instead of Algorithm 1.

VI. Experimental Results

To evaluate the proposed algorithm MDEMD, we have performed comparisons with the mean shift algorithm [9] and the standard DEMD tracking method [40], [41]. In the same context, we have also evaluated the Kalman based DEMD tracker (MDEMD-K). The proposed adapted Kalman filter is also combined with the mean shift (MS-K) and the DEMD algorithm (DEMD-K) in order to have a complete overview of its behavior. Six test sequences were employed in the evaluation consist of outdoor testing situations. The length of the sequences varies between 80 and 400 frames with one object to be tracked in every image sequence. Representative frames are shown in figure 2. Each object is described by its center, in image coordinates, and the size of the ellipse around it (the ellipse has axes parallel to the image axes). The ground truth in every image was determined manually. All the algorithms assume knowledge of the objects position only in the first frame. The object position is estimated in the following frames, using the corresponding algorithm. In all tests, the number of histogram bins for the mean shift is 16 and for the standard DEMD algorithms was 8 and 16 as suggested in [9], [40], [41]. The number of the components in the proposed GMM based tracker was selected to be 3, 4 and 6. The respective number of components may not be suitable for every problem and it depends on the complexity of the object to be tracked. As a rule of thumb, the number of components is equal to the colors of the object plus the colors of the background. For instance, by using three components, we assume that two components belong to the object (e.g. in *Seq5* a red car with black windows) and one component corresponds to the background (e.g. the gray road). All the examples were carried out with a core 2 Duo 1.66 GHz processor with 2GB RAM under Matlab.

Sequences *Seq1* and *Seq2* show a person walking from left to right in an underground station (PETS 2006 workshop). *Seq3* show a car moving (PETS 2001 workshop). *Seq4* shows a person walking in an outdoor environment (BEHAVE dataset, University of Edinburgh, School of Informatics, http://groups.inf.ed.ac.uk/vision/BEHAVEDATA/INTERACTIONS/). The last two sequences



Seq1 (129 frames) *Seq2* (80 frames) *Seq3* (400 frames)

Seq4 (221 frames) *Seq5* (106 frames) *Seq6* (285 frames)

Fig. 2. Representative frames of the image sequences used in the experiments.

(*Seq5* and *Seq6*) are created by our group. They show a red car moving from left to right without occlusion (*Seq5*) and with occlusion (*Seq6*).

To estimate the accuracy of the compared algorithms we measure the normalized Euclidean distance between the true center ($\mathbf{c}$) of the object (as determined by the ground truth) and the estimated location of the ellipse center ($\hat{\mathbf{c}}$). The normalized Euclidean distance is defined by

$$NED\left(\mathbf{c},\hat{\mathbf{c}}\right)=\sqrt{\left(\frac{\mathbf{c}_x-\hat{\mathbf{c}}_x}{h_x}\right)^2+\left(\frac{\mathbf{c}_y-\hat{\mathbf{c}}_y}{h_y}\right)^2} \tag{32}$$

where we recall that h_x and h_y are the ellipse dimensions. This implies that if $NED\left(\mathbf{c},\hat{\mathbf{c}}\right)<1$, then the estimated ellipse center $\hat{\mathbf{c}}$ is inside the ground truth ellipse. By these means, the image size and the ellipse dimensions do not influence the relative distance between ($\mathbf{c}$) and ($\hat{\mathbf{c}}$).

Table I summarizes the comparisons in terms of tracking accuracy. As it can be seen, the proposed algorithm has high accuracy (the computed center is inside the ellipse of the actual object). The standard DEMD algorithm confirms its efficiency with respect to mean shift, as it is presented in [40], [41]. Moreover, the proposed algorithm based on GMM is favorably compared with mean shift (table I). Generally, all of the compered methods present high performances with little differences. DEMD performs better at *Seq1* and *Seq3*, MDEMD is more accurate at *Seq2* and *Seq4* and all of the methods achieve similar results at *Seq5*. Also, the employment of Kalman filter has the ability to track objects when occlusions occur, while the other methods fail. When $NED>1$ the target is lost, which is the case for the methods not using the adapted Kalman filter (*Seq6*). Moreover, MDEMD-K with four components provides highly better accuracy with respect to MS-K and DEMD-K.

TABLE I

TRACKING ACCURACY. THE AVERAGE NORMALIZED EUCLIDEAN DISTANCE BETWEEN THE TRUE OBJECT CENTER AND THE ESTIMATED OBJECT CENTER IS PRESENTED FOR THE COMPARED METHODS.

Sequence	*Seq1*	*Seq2*	*Seq3*	*Seq4*	*Seq5*	*Seq6*
Frames	129	80	400	221	106	285
MS (16 bins)	0.44	0.38	0.46	4.01	0.07	2.08
DEMD (8 bins)	**0.16**	0.17	0.39	4.09	**0.05**	2.56
DEMD (16 bins)	0.17	0.16	**0.36**	0.41	**0.05**	2.50
MDEMD (3 components)	0.20	**0.15**	0.38	**0.38**	**0.05**	2.14
MDEMD (4 components)	0.23	**0.15**	0.40	0.51	0.06	1.98
MDEMD (6 components)	0.97	0.27	0.48	4.03	**0.05**	2.11
MS-K (16 bins)	0.55	0.47	0.48	5.47	0.09	0.72
DEMD-K (16 bins)	0.19	0.22	0.42	0.44	0.06	0.56
MDEMD-K (4 components)	0.25	0.25	0.39	0.94	**0.05**	**0.48**

As it can also be observed in table I when DEMD employs relatively few bins its results deteriorate in comparison with configurations using larger number of bins. The opposite stands for the proposed MDEMD. The is also confirmed by *Seq4* (table I) where DEMD with 8 bins totally misses the target (this is also true for MDEMD with 6 components). This is a relative difficult sequence as an indoor camera records a person moving from left to right outside. There is a glass between the camera and the moving person. This sequence has particular difficulties such as reflections due to the glass and illumination changes between frames. These difficulties are responsible for the failure of mean shift (MS) even when it is jointly applied with the adapted Kalman filter (MS-K).

The comparison of the three algorithms employing the Kalman filter (last three rows of table I) reveals that DEMD and MDEMD show similar accuracies (in any case they are better than mean shift).

In table II, the execution times (sec/frame) of the compared methods are shown. The proposed GMM based methods (MDEMD and MDEMD-K) are significantly faster with respect to DEMD (table II). This occurs because the number of GMM components is less than the number of histogram bins. Therefore, the integration of mixtures in DEMD tracking preserves the tracking accuracy and simultaneously reduces the computational complexity.

In table III we present the mean number of iterations needed for each method to converge in a single frame. The values of this table are independent from the machine used for the experiments and better highlight the rate of convergence of the various algorithms. As it can be seen the new

TABLE II

Average execution times for the compared methods (sec/frame).

Sequence	*Seq1*	*Seq2*	*Seq3*	*Seq4*	*Seq5*	*Seq6*
Frames	129	80	400	221	106	285
MS (16 bins)	1.64	2.91	0.66	0.55	8.05	0.36
DEMD (8 bins)	1.14	1.88	0.58	0.48	2.30	0.40
DEMD (16 bins)	2.42	3.82	1.39	1.42	4.67	2.45
MDEMD (3 components)	0.53	0.89	**0.23**	0.21	1.81	0.20
MDEMD (4 components)	**0.52**	0.77	0.24	0.20	1.71	0.22
MDEMD (6 components)	0.55	0.80	0.27	0.23	1.66	**0.17**
MS-K (16 bins)	2.19	3.85	0.68	0.64	10.53	0.38
DEMD-K (16 bins)	3.48	4.86	1.32	1.68	6.38	1.04
MDEMD-K (4 components)	0.58	**0.49**	0.24	**0.16**	**1.56**	0.28

TABLE III

Average number of iterations per frame for the compared methods (iterations/frame).

Sequence	*Seq1*	*Seq2*	*Seq3*	*Seq4*	*Seq5*	*Seq6*
Frames	129	80	400	221	106	285
MS (16 bins)	2.98	3.86	**1.00**	1.90	2.17	**1.10**
DEMD (8 bins)	4.76	4.90	2.40	3.46	3.36	2.78
DEMD (16 bins)	4.71	4.82	2.56	3.77	3.46	2.77
MDEMD (3 components)	2.43	2.62	1.23	2.34	1.88	1.51
MDEMD (4 components)	2.27	2.22	1.27	2.10	1.75	1.39
MDEMD (6 components)	**2.05**	2.05	1.17	1.95	1.67	1.29
MS-K (16 bins)	3.86	5.31	1.03	2.37	2.93	1.34
DEMD-K (16 bins)	6.57	7.15	2.46	4.87	4.54	2.89
MDEMD-K (4 components)	2.58	**1.48**	1.54	**1.66**	**1.44**	1.30

MDEMD algorithm converges in fewer iterations than the standard DEMD. Let us also notice that DEMD converges in approximately the same number of iterations with 8 and 16 bins. However the average execution time per frame (as depicted in table II) is almost doubled when using 16 bins. This is due to the augmented complexity in the optimization algorithm. Furthermore, the application of the adaptive Kalman filter reduces the number of iterations if the motion model is correctly estimated.

A representative example of *Seq5* is shown in figure 3. An example with occlusion is presented in figure 4 where the red car is masked by the trees. All of the compared algorithms successfully track the object until it reaches the trees. However, only the algorithms employing the proposed adaptive

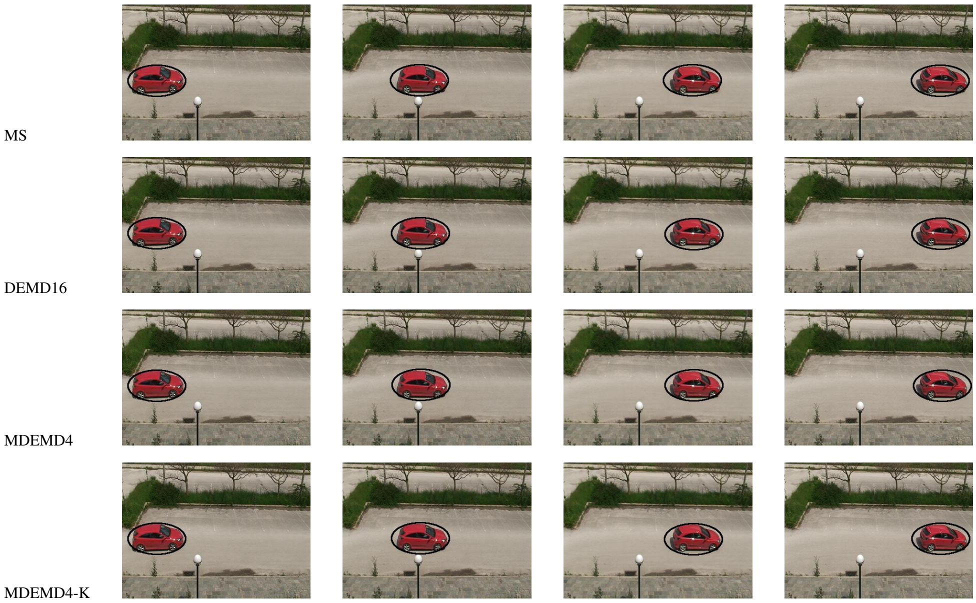


Fig. 3. *Seq5*. Representative frames and the normalized Euclidean distances between the ground truth and the estimates of the ellipse center for the compared algorithms.

Kalman filter achieve in predicting its motion.

VII. Conclusions

We have proposed a method for visual object tracking relying on modeling the appearance of the object in the first frame using a Gaussian mixture. The EM algorithm is applied to compute the initial GMM. In the following frames, the location of the object is estimated in a differential framework by the direction of the gradient of the EMD with respect to the bi-dimensional image space [40], [41]. This gradient is computed in closed form and the key issue in this computation, is the change in the responsibilities of the GMM components between adjacent frames. In these images the EM is not applied, because means and variances do not change. Therefore, the algorithm is significantly faster than the standard DEMD tracker [40], [41] while retaining the same high tracking accuracy. Also, the proposed algorithm is combined with a Kalman filter to efficiently handle occlusions. The prediction of the GMM-based DEMD tracker is considered as the observation of a Kalman filter whose state parameters are automatically determined based on recent motion history. By these means, partial or total occlusions may be successfully addressed. Future work consists in considering multiple target tracking, as in that case, the current algorithm has to be executed independently for each object. Also, other probabilistic modeling approaches could be employed for the estimation of the parameters of the state matrix of the Kalman filter.

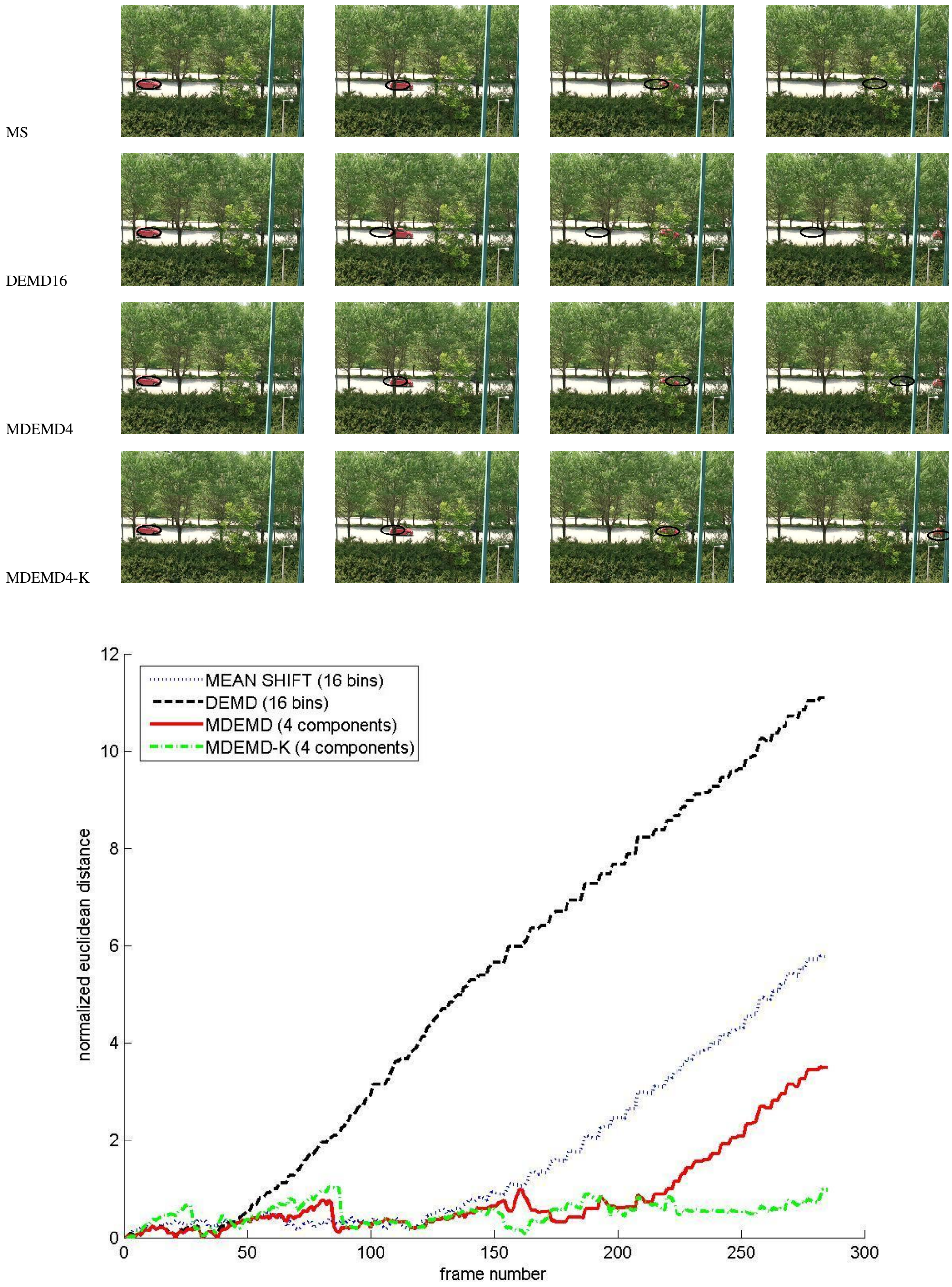


Fig. 4. *Seq6*. Representative frames and the normalized Euclidean distances between the ground truth and the estimates of the ellipse center for the compared algorithms.

References

[1] A. A. Argyros and M. I. A. Lourakis. Real-time tracking of multiple skin-colored object with a possibly moving camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 368–379, 2004.

[2] S. Arulampalam, S. Maskell, and N. Gordon. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[3] S. Avidan. Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(8):1064–1072,

2004.

[4] D. Beymer and K. Konolige. Real-time tracking of multiple people using continuous detection. In *Proceedings of International Conference on Computer Vision (ICCV99)*, 1999.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[6] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, 26:63–84, 1998.

[7] A. Bugeau and P. Perez. Track and cut: Simultaneous tracking and segmentation of multiple objects with graph cuts. *EURASIP Journal on Image and Video Processing*, 2008(ID:317278), 2008.

[8] E. K. P. Chong and S. H. Zak. *An Introduction to Optimization*. Wiley-Interscience Series in Discrete Mathematics and Optimization, 3rd edition, 2008.

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

[10] D. Cremers. Dynamical statistical shape priors for level set based tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1262–1273, 2006.

[11] E. Cuevas, D. Zaldivar, and R. Rojas. Kalman filter for vision tracking. Technical Report B 05-12, Freier Universitat Berlin, Institut fur Informatik, 2005.

[12] Z. Fan, M. Yang, and Y. Wu. Multiple collaborative kernel tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7):1268–1273, 2007.

[13] M. A. T. Figueiredo and A. K. Jain. Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:381–396, 2000.

[14] G. D. Hager, M. Dewan, and C. V. Stewart. Multiple kernel tracking with SSD. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR04)*, 1:790–797, 2004.

[15] B. Han, S.-W. Joo, and L. S. Davis. Probabilistic fusion tracking using mixture kernel-based Bayesian filtering. In *Proceedings of International Conference on Computer Vision (ICCV07)*, pages 1–8, 2007.

[16] M. Isard and A. Blake. Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28, 1998.

[17] M. Isard and A. Blake. Icondensation - unifying low-level and high-level tracking in a stochastic framework. In *Proceedings of the 5th European Conference on Computer Vision (ECCV)*, pages 893–908, 1998.

[18] D. G. Lowe. Distinctive image features from Scale-Invariant keypoint. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[19] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, pages 674–679, 1981.

[20] A. Mansouri. Region tracking via level set pdes without motion computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:947–961, 2002.

[21] F. Moreno-Noguer, A. Sanfeliu, and D. Samaras. Dependent multiple cue integration for robust tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(4):670–685, 2008.

[22] D. P. Mukherjee, S. Member, N. Ray, S. T. Acton, and S. Member. Level set analysis for leukocyte detection and tracking. *IEEE Transactions on Image Processing*, 13:562–572, 2004.

[23] N. Paragios and R. Deriche. Geodesic active contours and level sets for the detection and tracking of moving objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:266–280, 2000.

[24] Y. Rathi, N. Vaswani, A. Tannenbaum, and A. Yezzi. Particle filtering for geometric active contours with application to tracking moving and deforming objects. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2005 (CVPR 2005)*, 2:2–9, 2005.

[25] Y. Rubner, C. Tomasi, and L. J. Guibas. The earth mover's distance as a metric for image retrieval. *International Journal of Computer Vision*, 40(2):99–121, 2000.

[26] K. Sato and J. K. Aggarwal. Temporal spatio-velocity transform and its application to tracking and interaction. *Computer Vision and Image Understanding*, 96(2):100–128, 2004.

[27] J. Shi and C. Tomasi. Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR94)*, pages 593–600, 1994.

[28] D. Simon. *Optimal state estimation: Kalman, H_∞ and non linear approaches.* Wiley–Interscience, 2006.

[29] C. Stauffer and W. E. L. Grimson. Learning patterns of activity using real-time tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):747–757, 2000.

[30] H. Tao, H. S. Sawhney, and R. Kumar. Object tracking with Bayesian estimation of dynamic layer representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):75–89, 2002.

[31] J. Tu, H. Tao, and T. Huang. Online updating appearance generative mixture model for Mean-Shift tracking. *Machine Vision and Applications*, 20(3):163–173, 2009.

[32] C. J. Veenman, M. J. T. Reinders, and E. Backer. Resolving motion correspondence for densely moving points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(1):54–72, 2001.

[33] H. Wang, D. Suter, and K. Schindler. Effective appearance model and similarity measure for particle filtering and visual tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 606–618, 2006.

[34] Z. Wang, X. Yang, Y. Xu, and S. Yu. Camshift guided particle filter for visual tracking. *Pattern Recognition Letters*, 30(4):407–413, 2009.

[35] Y. Wu and T. S. Huang. Robust visual tracking by integrating multiple cues based on co-inference learning. *International Journal of Computer Vision*, 58(1):55–71, 2004.

[36] C. Yang, R. Duraiswami, and L. Davis. Efficient mean-shift tracking via a new similarity measure. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR05) - Volume 1*, pages 176–183, 2005.

[37] M. Yang, Y. Wu, and G. Hua. Context-aware visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(7):1195–1209, 2009.

[38] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4):1–45, 2006.

[39] A. Yilmaz, X. Li, and M. Shah. Contour-based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(11):1531–1536, 2004.

[40] Q. Zhao, S. Brennan, and H. Tao. Differential EMD tracking. In *Proceedings of International Conference on Computer Vision (ICCV07)*, pages 1–8, 2007.

[41] Q. Zhao and H. Tao. Differential Earth Mover's Distance with its application to visual tracking. *to appear in IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2009.

[42] H. Zhoo, Y. Yuan, and C. Shi. Object tracking using SIFT features and mean shift. *Computer Vision and Image Understanding*, 113(3):345–352, 2009.