Visual Object Tracking based on Local Steering Kernels and Color Histograms

Olga Zoidi, Anastasios Tefas, Ioannis Pitas

Abstract— **In this paper, we propose a visual object tracking framework, which employs an appearance-based representation of the target object, based on Local Steering Kernel Descriptors and color histogram information. This framework takes as input the region of the target object in the previous video frame and a stored instance of the target object and tries to localize the object in the current frame by finding the frame region which best resembles the input. As the object view changes over time, the object model is updated, hence incorporating these changes. Color histogram similarity between the detected object and the surrounding background is employed for background subtraction. Experiments were conducted to test the performance of the proposed framework in various conditions. The proposed tracking scheme was proven to be successful in tracking objects under scale and rotation variations and partial occlusion, as well as in tracking rather slowly deformable articulated objects.**

I. Introduction

Visual tracking of an object in an image sequence is important for many applications, such as automatic video surveillance [1], autonomous robotic systems [2], human-computer interfaces [3], augmented reality [4] and e-healthcare [5]. However, this task is difficult to accomplish, as, in real life situations, the illumination conditions may vary and the object may be non-rigid or articulated, or occluded by background objects, and/or it may perform rapid and complicated movements, hence deteriorating tracking performance. In order to solve the above mentioned problems, numerous tracking algorithms have been proposed, which employ techniques for object representation (based on object features, texture and shape models, or object contours), object position prediction and search in the next video frame. The object representation methods can be divided into five categories [6]: model-based, appearance-based, contour-based, feature-based and hybrid ones.

Model-based tracking methods exploit a priori information about the object shape, creating a $2D$ or $3D$ model for the object [7]. These methods can address the problem of object tracking under illumination variations, changes in the object viewing angle and partial occlusion. However, their computational cost is heavy, especially when tracking objects with complex $3D$ geometry. Moreover, they require the implementation of a detailed model for each type of object in the scene, as the models cannot be easily generalized. Appearance-based tracking methods use the visual information of the object projection on the image plane, i.e., color, texture and shape, as well as information on the $2D$ object motion [8]. These methods deal with simple object transformations, such as affine ones, including translation and rotation. However, they are sensitive to illumination changes. Contour-based tracking methods track the object contour by employing shape-matching or contour-evolution techniques [9]. Contours can be represented by active models, such as snakes, B-splines and geodesic active contours, or meshes [10], enabling the tracking of both rigid and non-rigid objects. In order to deal with partially occluded objects, tracking algorithms incorporate occlusion detection and estimation techniques. Feature-based methods perform object tracking by tracking a set of feature points, which represent the object [11]. These tracked features are then grouped, according to their associations in the previous frame. These methods perform well in partial occlusion, as well as in tracking very small objects. The major problem of feature-based methods is the correct distinction between the target object and background features. Finally, hybrid methods for object tracking exploit the advantages of the above mentioned methods, by incorporating two or more tracking methods [12]. Usually, feature-based methods are employed first, for object detection and localization. Then, region-based techniques are used to track its parts. The main disadvantage of these methods is their high computational complexity.

Instead of targeting our efforts to make a model for the target object and then find its location in the video, we can address the dual problem: export a model for the scene, called background and then find the regions in the video frame which deviate from the background model. Regions with high deviation from the model imply the existence of a moving object. Such methods perform object tracking e.g. with background subtraction [13], [14]. They are computationally efficient and can model illumination changes, noise and periodic movements. However, they can be applied only on static scenes obtained from stationary cameras or scenes with small motion obtained from portable cameras, as camera movement distorts the background model or may imply the use of multiple background models.

Most of the object tracking algorithms in bibliography, including the proposed tracking scheme, employ an appearance-based object representation. The earlier methods consider an almost constant appearance model for the object which is

O. Zoidi, A. Tefas and I. Pitas are with the Department of Informatics, Aristotle University of Thessaloniki, Thessaloniki 54124, Greece. e-mail: {ozoidi,tefas,pitas}@aiia.csd.auth.gr.

The research leading to these results has received funding from the Collaborative European Project MOBISERV FP7-248434 (http://www. mobiserv.eu), An Integrated Intelligent Home Environment for the Provision of Health, Nutrition and Mobility Services to the Elderly.

extracted from the object initialization in the first video frame and, therefore, it does not change over time. As a result, these methods cannot handle severe changes in the object view and, in some cases, partial occlusion. Example of such methods are the Mean Shift (MS) tracking algorithms [15], [16], [8] and [17], which use variations of the MS algorithm [18], in order to detect the candidate object with the most similar color histogram to the target object. The problem of partial occlusion in appearance-based tracking schemes has been addressed by decomposing the target object into non-overlapping [19] or overlapping [20] fragments, which are tracked separately. The fragments can be selected either manually, or randomly. The number and size of the fragments play an important role in tracking performance, as too many or too big fragments result in heavy computational weight and, on the contrary, too few fragments cause the tracker to drift. The new position of the object can be estimated by various voting techniques for the confidence of each fragment, e.g. by the fragment with the maximum confidence, or by selecting the smaller area which contains all the fragment tracking results. The changes in the object view angle are handled by either multiple hypotheses for the object state [21], or by considering adaptive appearance models [22]. These methods are based on the sequential Monte Carlo method, also known as Particle Filters [23]. Other approaches employ a hierarchical framework based on bounded irregular pyramids [24] and an incremental eigenbasis learning framework [25].

Our tracking approach is an appearance based one using both the color histograms to describe object color information and the Local Steering Kernel (LSK) object texture descriptors [26]. A preliminary work on visual object tracking based on LSKs was presented in [27]. We first search image regions in a video frame that have high color similarity to the object color histogram. Once these candidate regions are found, the illumination-invariant Local Steering Kernel (LSK) descriptors [26] of both the target object and the candidate search region are extracted. LSKs are descriptors of the image salient features. They were first employed as an image denoising and reconstruction technique [28] and later found application in object detection [26]. As an object detector, they were proven to be robust in small scale and orientation changes, as well as small object deformations. Therefore, their incorporation in a tracking framework results in successful tracking of slowly deformable objects. After discarding the image regions with small color histogram similarity to the object color histogram, the new position of the object is selected as the image region, whose LSK representation has the maximum similarity to the one of the target object. As tracking evolves, every time the target object appearance changes, either due to rotation/zooming, or a deformation, or a change in the view angle, the object model, being a stack containing different instances of the object including information about its scale and 2-dimensional angle, is updated with the representation of the most recent detected object instance. This way, the algorithm is able to cope with changes in object appearance. The final decision on the new tracked object location is determined to be the candidate image region with the maximal average LSK similarity to the detected object instance in the previous frame and the most recent instance in the object model (stack). As proven in experiments, the overall tracking algorithm succeeds in illumination-invariant tracking of rigid objects with severe changes in view angle, or being subject to affine transformations and/or partial occlusion. The novelties of the proposed approach are:

- the use of online-training of the object model (stack) based on LSKs,
- the use of an efficient framework for scale, rotation and location adaptive tracking combined with LSKs,
- the combination of LSKs with color histogram of candidate object regions for enhanced tracking performance.

The remaining of the paper is organized as follows. Section II-A presents the use of color information in tracking, essentially for performing background subtraction. Section II-B presents the LSK feature extraction method. Section II-C describes the method for deciding on the object position in the video frame under investigation and the algorithm for updating the object model. Section II-D describes the method for extracting the search area for the new position of the object in the next frame. Section III presents the experimental results and evaluation of the proposed method, compared with two other state of the art tracking algorithms. Finally, conclusions are drawn in Section IV.

II. LSK Object Tracking

In this paper, we propose a novel appearance-based method for tracking both rigid and deformable objects in a video, without prior object appearance model training. The proposed framework makes the assumption that object translation and deformation between two consecutive video frames is rather small. Each transformation of the object image, i.e., scaling due to zooming or rotation, is considered as an object instance and it is stored in a stack, i.e., a list of object instances (images). The stored object instances comprise the object model. As tracking evolves, the object model is updated with new object instances, incorporating the transformations the object undergoes.

In each new video frame, the new object Region of Interest (ROI) is searched in a local region around a predicted object position, called search region. The search region may contain several candidate object ROIs in the new video frame. The algorithm employs spatial information through Local Steering Kernels (LSKs) [26] and color information through color histogram (CH) for representing both the object instances and the search region. The similarity of the object salient spatial features and color histogram between a candidate object ROI and the object region in the previous frame and the last updated object instance from the object model (stack) are evaluated. The cosine similarity of the object salient features (i.e., LSK descriptors) is robust to small object appearance changes between two consecutive video frames. In each frame, the patch of the search region with the maximum average LSK similarity to the object image in the previous frame and a stored object instance in the object appearance model is selected as the new object instance. The drop of the maximum average LSK similarity at the current video frame under a threshold, which is determined with respect to the maximal average similarity at the previous video frame, indicates that the object appearance changed. This change is embedded in the tracking framework, by storing the detected object instance in the object appearance model. In the next frame, the search region patches will be compared to the last stored object instance. Thus, the proposed tracking framework is able to follow changes in the object appearance, due to view point alterations and/or object movement or deformations.

The proposed method consists of the following steps:

- Initialization of the object ROI in the first video frame. The initialization can be done either manually, by selecting a bounding box around the object we want to track, or automatically, using an object detection algorithm, e.g. the one based on LSKs [26].
- Color similarity search in the current search region, using color histogram information, which essentially leads to background subtraction and reduction of the number of the candidate object ROIs.
- Representation of both the object and the selected search region through their salient features that are extracted using Local Steering Kernels (LSKs).
- Decision on the object ROI in the new video frame, based on the measurement of the salient feature similarities between a candidate object ROI and a) the object ROI in the previous frame and b) the last stored object instance in the object model (stack) and finding a match.
- Update the object model by storing its different views (called object instances) in a stack. When the match is successful, this update is done by pushing a new object instance in the stack, when the object undergoes an affine transformation, i.e., scale and rotation, or changes view.
- Prediction of the object position in the following video frame and initialization of an object search region. The position prediction is based on the assumption that the object performs rather smooth motion.

A. Color Similarity

In order to discriminate the object from its background, we can exploit their color difference, by histogram comparison. In most cases, the object color histogram does not remain the same. On the contrary, it is sensitive to changes in illumination, as well as to view point changes. After object position prediction and search region selection, the search region of size $R_1 \times R_2$ is divided into candidate object ROIs (patches) shifted by d pixels vertically and horizontally and having size equal to the size of the query object $Q_1 \times Q_2$. In total, the number of created patches is $\frac{R_1-Q_1+1}{d} \times \frac{R_2-Q_2+1}{d}$. The parameter $1/d$ determines the density of the uniformly selected candidate object ROIs. By setting $d = 1$, the maximum number of possible candidate object ROIs in the search region is selected, which essentially leads to exhaustive search of the object in the search region. The increase of the value of d is, essentially, a uniform sampling of the candidate object ROIs every d pixels in the search region. At frame t, the $B_t\%$ of the search region patches with the minimal histogram similarity to the object histogram are considered to belong to the background. It has to be noted that B_t is not constant throughout tracking, but it is computed at each frame t, as we shall show later on in the section. For each image patch we extract three color histograms, one for each R, G and B component. The color histograms are compared according to cosine similarity. Instead of the cosine similarity, other more sophisticated metrics can be used, such as the Bhattacharyya distance. Cosine similarity was chosen because it consists a good compromise between low computational cost and performance, as proven experimentally. The cosine similarity between two histograms $\mathbf{h}_1$, $\mathbf{h}_2 \in \mathbb{R}^{256}$ is given by:

$$c(\mathbf{h}_1, \mathbf{h}_2) = \cos(\theta) = \frac{< \mathbf{h}_1, \mathbf{h}_2 >}{\|\mathbf{h}_1\|\|\mathbf{h}_2\|}, \tag{1}$$

where $< \cdot >$ defines the inner product of $\mathbf{h}_1$, $\mathbf{h}_2$, θ denotes the angle they form and $\|\cdot\|$ denotes the Euclidean norm. The cosine similarity takes values in the range $[-1, 1]$. In order to map the range $[-1, 1]$ to $[0, \infty)$, we apply the transformation $S = c^2/(1 - c^2)$ [26].

The final similarity measure between the two color regions is computed by summing the transformed cosine similarity measures S for the three color channels. The similarity values of all patches comprise a matrix of color histogram similarity $\mathbf{M}_{CH}$.

Color histogram (CH) similarity is an indicator of whether the search region patch belongs to the object ROI or the background. We expect patches with lower CH similarity to belong to the background and patches with higher CH similarity to belong to the object. Therefore, by exploiting CH information, we can find a proper threshold τ_t to exclude search region patches, which belong to the background, from being considered part of the candidate object ROIs. The threshold τ_t is computed for each frame t and it depends on the CH similarity distribution $p(M_{ij})$ of matrix $\mathbf{M}_{CH}$ entries M_{ij}, $i = 1, \ldots, \frac{R_1-Q_1+1}{d}$, $j = 1, \ldots, \frac{R_2-Q_2+1}{d}$ at frame t. If the background color is significantly different from that of the object color, the distribution of $\mathbf{M}_{CH}$ takes small values and we set τ_t so as to achieve a high confidence level $B_t\%$ in deciding whether the patch under consideration is a valid candidate object ROI. At frame t, the confidence level $B_t\%$ decreases in the case where the background color is similar to the object color and the majority of the $\mathbf{M}_{CH}$ entries take high values. Setting $\bar{M}$, M_{max} and M_{min} as the mean, maximal and minimal values of $\mathbf{M}_{CH}$ entries, respectively, we estimate the confidence level $B_t\%$ as follows:

$$B_t = 100 \cdot \begin{cases} 1 - \frac{|\bar{M} - M_{max}|}{|\bar{M} - M_{min}|}, & \text{if } |\bar{M} - M_{max}| < |\bar{M} - M_{min}|, \\ \frac{1}{2}, & \text{if } |\bar{M} - M_{max}| = |\bar{M} - M_{min}|, \\ \frac{|\bar{M} - M_{min}|}{|\bar{M} - M_{max}|}, & \text{if } |\bar{M} - M_{max}| > |\bar{M} - M_{min}|. \end{cases} \tag{2}$$

The threshold τ_t is computed for each video frame t as the matrix $\mathbf{M}_{CH}$ entry, which is less or equal than $B_t\%$ of the $\mathbf{M}_{CH}$ entries. Finally, we compute the binary matrix $\mathbf{B}_{CH}$, whose (i, j)-entry is set to 1, if the (i, j)-entry of $\mathbf{M}_{CH}$ is greater of equal than τ_t and 0, otherwise. The use of this matrix in tracking is detailed in section II-C.

B. Object texture description

Edges carry important image texture information that can be used for a good image representation. Various methods exist to this end, such as Gabor filters [29], Difference of Gaussian (DoG) filters, or even simple luminance gradient estimation methods (Sobel filters, Prewitt filters) [30]. In the proposed framework, texture image representation is performed with Local Steering Kernels (LSKs) [26]. LSKs are local descriptors of the image structure, which exploit both spatial and pixel-value information. They are a non-linear combination of weighted spacial distances between a pixel of an image of size $N_1 \times N_2$ and its surrounding $M \times M$ pixels. The distance between an image pixel $\mathbf{p}$ and its neighboring pixel $\mathbf{p}_i$ is measured using a weighted euclidean distance, which uses as weights the covariance matrix $\mathbf{C}_i$ of the image gradients along x (horizontal) and y (vertical) axes:

$$K_i(\mathbf{p}) = \frac{\sqrt{\det(\mathbf{C}_i)}}{2\pi} \exp\left\{-\frac{(\mathbf{p}_i-\mathbf{p})^T\mathbf{C}_i(\mathbf{p}_i-\mathbf{p})}{2}\right\}, i=1,\ldots,M^2, \tag{3}$$

where $\mathbf{p} = [x, y]^T$ are the pixel coordinates. It is known that the covariance matrix $\mathbf{C}_i$ of these gradients contains information about the dominant edge orientation in a local image region, described by the eigenvector, which corresponds to the eigenvalue with the largest absolute value. Therefore, the covariance matrix is used to rotate, elongate and scale the Gaussian kernel along the local edge. In order to estimate the $\mathbf{C}_i$ matrix in (3), we calculate the gradients $\mathbf{g} = \nabla\mathbf{f}(\mathbf{p}) = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]^T$ of the image $f(\mathbf{p})$ along x and y axes and we measure their correlation in a neighbor of $M \times M$ pixels centered at the pixel $\mathbf{p}_i = [x_i, y_i]^T$. The gradient vectors $\mathbf{g}_i$, $i = 1, \ldots, M^2$ in the neighbor $M \times M$ are column-stacked in matrix $\mathbf{G}_i$:

$$\mathbf{G}_i = \begin{bmatrix} g_{x1} & g_{y1} \\ g_{x2} & g_{y2} \\ \vdots & \vdots \\ g_{xM^2} & g_{yM^2} \end{bmatrix}. \tag{4}$$

The correlation matrix $\mathbf{C}_i$ is calculated via the Singular Value Decomposition (SVD) of $\mathbf{G}_i$ [31]:

$$\mathbf{G}_i = \mathbf{U}_i\mathbf{S}_i\mathbf{V}_i^T = \mathbf{U}_i \begin{bmatrix} s_{1i} & 0 \\ 0 & s_{2i} \end{bmatrix} \begin{bmatrix} \mathbf{v}_{1i}^T \\ \mathbf{v}_{2i}^T \end{bmatrix} \tag{5}$$

$$\alpha_1 = \frac{s_{1i}+1}{s_{2i}+1}, \quad \alpha_2 = \frac{1}{\alpha_1}, \quad \gamma = \left(\frac{s_{1i}s_{2i}+10^{-7}}{M^2}\right)^{0.008} \tag{6}$$

$$\mathbf{C}_i = \gamma\left(\alpha_1^2\mathbf{v}_{1i}\mathbf{v}_{1i}^T + \alpha_2^2\mathbf{v}_{2i}\mathbf{v}_{2i}^T\right), \tag{7}$$

where s_{1i} and s_{2i} are the singular values of $\mathbf{G}_i$ and $\mathbf{v}_{1i}^T$, $\mathbf{v}_{2i}^T$ are the corresponding singular vectors. For an image pixel $\mathbf{p}$, equation (3) is computed for each neighboring pixel $\mathbf{p}_i$, $i = 1, \ldots, M^2$, meaning that for each image pixel we extract an LSK vector $\mathbf{K}(\mathbf{p}) \in \Re^{M^2 \times 1}$. In order for the image representation to be invariant to illumination changes, we normalize the LSK vectors:

$$\mathbf{K}_N(\mathbf{p}) = \frac{\mathbf{K}(\mathbf{p})}{|\mathbf{K}(\mathbf{p})|_1}, \tag{8}$$

where $|\cdot|_1$ is the L_1-norm. The LSK vectors of each image pixel are then transposed and ordered column-wise into the matrix $\mathbf{Q} \in \Re^{N_1N_2 \times M^2}$.

LSKs are good image texture descriptors, because they are invariant to brightness variations, contrast changes and noise. In our approach, first the object ROI and the search region are converted from the RGB to the La^*b^* color space and, then, the LSKs are computed for each color channel separately, through equations (3)-(8). The final representation for the object ROI comprises its salient characteristics and is obtained by applying PCA to retain 80% of the information in the LSK kernels [26]. The resulting projection matrix will then be used for the dimensionality reduction of the LSK descriptors of the search region. Finally, the search region is divided into patches and the LSK similarity matrix is estimated by applying the cosine similarity measure, according to the procedure described in section II-A.

C. Object localization and model update

Object localization in the search region is performed by taking into account LSK and color histogram similarity of a candidate object ROI (patch) to the object ROI in the previous frame and the last stored object instance in the object model (stack). More specifically, we divide the object search region into overlapping patches of size equal to that of the detected object and, for each patch, we extract the LSK and CH features as described in Subsections II-B and II-A. Then, for each patch, we construct three cosine similarity matrices, two for the LSK resemblance between this patch and a) the detected object in the previous frame and b) the last updated object instance and one for the CH similarity between this patch and the last stored object instance. The new object ROI is the candidate region with the maximal mean LSK similarity to the object ROI in the previous frame and the last stored object instance. The final decision matrix is computed by:

$$\mathbf{M} = [(1-\lambda)\mathbf{M}_{LSK1} + \lambda\mathbf{M}_{LSK2}] * \mathbf{B}_{CH}, \tag{9}$$

where $0 \leq \lambda \leq 1$ is a suitably chosen weight, $\mathbf{M}_{LSK1}$, $\mathbf{M}_{LSK2}$ are the LSK similarity matrices for the last detected object and the last object instance, respectively, $\mathbf{B}_{CH}$ is the binary CH similarity matrix, and $*$ denotes the element-wise matrix multiplication. λ usually takes the value 0.5. The reason why we take into account the similarity with the last updated object instance is that it prevents the tracker from drifting, when the object is partially occluded. The new candidate object position is at the patch having the maximal value $\max_{i,j}(\mathbf{M}_{ij})$. We compare this value with the same maximal value for the detected object in the previous video frame. If the value drops under a predefined threshold T, it indicates a possible change in the object appearance, either because of a $2D$ affine transform (when the object image is rotated or scaled due to zooming), or because of a change in the object view angle. In order to determine the cause of the similarity drop, we search the search region for scaled and rotated versions of the object as follows.

In order to detect whether the object is rotated by $\pm\varphi$ degrees, we rotate the video frame t around the center of the predicted object position $\hat{\mathbf{p}}_t$ by $\mp\varphi$ degrees respectively, obtain the new search regions and calculate two decision matrices according to (9). Object scaling due to zooming is detected by resizing the search region by $\pm s\%$. When interpolation is needed, the new pixel values of the resized search region are estimated by bilinear interpolation. We note that, in both cases (rotation and scaling), the query object is left intact, which means that its representation through LSKs does not change. In order to find a robust search step for rotation and scaling, we have conducted several experiments, so that the robustness of the LSKs similarity under rotation and/or scaling is checked. In our experiments, we set the rotation step φ to 10 degrees and the scale step size s to 10%. In total, we examine four affine transformations, i.e., clockwise rotation, counter-clockwise rotation, up-scaling and down-scaling, plus the identity transformation. For each case, a new decision matrix is produced, according to (9). In order to ensure that the affine transformation of the object is detected correctly, the final decision for the new object is the one which corresponds to the maximal value of the five decision matrices, under the condition that the mean value of the corresponding decision matrix is greater than the mean value of the identity transformation decision matrix. Otherwise, the new object location at frame t is the position which corresponds to the maximal value of the identity transformation decision matrix. The newly localized object is stored in a stack. The stack size is constant throughout tracking and, in our experiments, was set to 5 object instances. When the stack is full, the oldest object instance is replaced by the new one. In the next video frame, the object will be searched at the scale and orientation values related to the last stored object instance.

D. Search region extraction in the next frame

After determining the object position in the current frame, the position of the object in the following frame is predicted using a linear Kalman filter. It is an iterative process for estimating the state $\mathbf{x}_t \in \Re^n$ of a discrete-time process in time t, given measurements $\mathbf{z}_t \in \Re^m$, both subject to white Gaussian noise [32]. In our system, the object motion state estimation model is given by $\mathbf{x}_t = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{w}_{t-1}$, where the state $\mathbf{x}_t = [x, y, dx, dy]^T \in \Re^8$ consists of the x and y object center coordinates and the object translation dx, dy. $\mathbf{w}_{t-1}$ denotes the process noise, with probability distribution $p(\mathbf{w}) \sim N(\mathbf{0}, \mathbf{Q})$, where $\mathbf{Q} \in \Re^{4\times 4}$ is the noise covariance matrix. $\mathbf{A} \in \Re^{4\times 4}$ is the transition matrix of the system:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{10}$$

The measurement $\mathbf{z}_t = [x, y]^T \in \Re^2$ is related to the state $\mathbf{x}_t$ with $\mathbf{z}_t = \mathbf{H}\mathbf{x}_t + \mathbf{v}_t$, where $\mathbf{H} \in \Re^{2\times 4}$ is the measurement matrix:

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

and $\mathbf{v}_k$ is the measurement noise, with $p(\mathbf{v}) \sim N(\mathbf{0}, \mathbf{R})$, where $\mathbf{R} \in \Re^{2\times 2}$ is the noise covariance. The estimation of the current state $\hat{\mathbf{x}}_t$ and the covariance matrix $\hat{\mathbf{P}}_t$ of the stochastic model are estimated through the set of equations:

$$\begin{aligned} \hat{\mathbf{x}}_t &= \mathbf{A}\hat{\mathbf{x}}_{t-1} \\ \hat{\mathbf{P}}_t &= \mathbf{A}\hat{\mathbf{P}}_{t-1}\mathbf{A}^T + \mathbf{Q}, \end{aligned} \tag{11}$$

while the stochastic model is adjusted through equations:

$$\begin{aligned} \mathbf{K}_t &= \hat{\mathbf{P}}_t\mathbf{H}^T(\mathbf{H}\hat{\mathbf{P}}_t\mathbf{H}^T + \mathbf{R})^{-1} \\ \hat{\mathbf{x}}_t &= \hat{\mathbf{x}}_{t-1} + \mathbf{K}_t(\mathbf{z}_t - \mathbf{H}\hat{\mathbf{x}}_t) \\ \mathbf{P}_t &= (\mathbf{I} - \mathbf{K}_t\mathbf{H})\mathbf{P}_t. \end{aligned} \tag{12}$$

The matrix $\mathbf{K}_t$ is called the Kalman gain and is chosen such that minimizes the a posteriori error covariance $\mathbf{P}_t$.

The object will then be searched in a search region centered at the predicted position $\hat{\mathbf{x}}_t$. The size of this region varies according to the expected maximal object velocity, the object size and the reliability of the predicted position. If the object moves fast, or it moves in a non-smooth trajectory, or it is large and we are not confident on our prediction, we select a large search region. In our experiments, if the object ROI size is $Q_1 \times Q_2$ pixels, then the search region size is set to $R_1 \times R_2 = 2Q_1 \times 2Q_2$ pixels. The object ROI dimensions $Q_1 \times Q_2$ are selected to be small to increase tracking speed but large enough in order to preserve the object salient features. Typical values of $Q_1 \times Q_2$ are around 30×30 pixels. The tracking procedure continues by repeating the steps described in sections II-A to II-C.

The linear Kalman filter is a rather simple but efficient method for motion state prediction. The efficiency of the tracking algorithm could be improved if the Kalman filter is substituted by more accurate non-linear Bayesian filters, such as the extended Kalman filter, Hidden Markov model (HMM) filters and particle filters [33], or by the mean shift algorithm [34].

III. Experimental Results

A. Experimental setup

The effectiveness of the proposed tracking scheme was tested in a number of videos coming from various applications. The initialization in each video was performed through a training-free object detection algorithm over the entire first video frame [26]. The search region size is $R_1 \times R_2 = 2Q_1 \times 2Q_2$, where $Q_1 \times Q_2$ are the downscaled object dimensions, which are selected for each experiment as shown in the third column of Table I.

The window size for the LSK feature extraction is 3×3 pixels. The rotation step is 10 degrees, except for some cases, where we do not expect $2D$ rotation of the tracked object and we set it equal to zero (e.g., when we track people by using surveillance cameras). The scale step is set to 10%. The threshold for the model update T is zero, which means that, every time the similarity value decreases, we search for possible scale and rotation of the object. Finally, the noise covariance matrix $\mathbf{Q}$ was set to the identity matrix $\mathbf{Q} = \mathbf{I} \in \Re^{4\times 4}$ and the value of the measurement noise covariance matrix $\mathbf{R}$ was set to the identity matrix $\mathbf{R} = \mathbf{I} \in \Re^{2\times 2}$. This initialization was proven to provide good tracking results.

B. Qualitative evaluation

The performance of the proposed tracker is compared with two other state of the art trackers found in the literature: one that incorporates an appearance based object representation (i.e., image intensity) with particle filters (called PF tracker) [22] and another one that performs object tracking by dividing the object of interest in smaller fragments (called FT tracker) [19]. The FT tracker is publicly available at the authors' site. In order to have a better understanding of the optimal tracking results for each tracking scheme, each tracker is initialized with a different object ROI that fits best to the characteristics of the tracking algorithm. For example, the PF tracker is initialized inside the tracked object, while the FT tracker and the CH-LSK trackers are initialized in such a way that the initial object ROIs contain the object boundary and a small amount of background. A summary of the main characteristics of the videos used in the experiments is shown in Table I.

a) Case studies 1 and 2: In the first two experiments, we test the performance of the proposed tracking scheme under variations of the object scale. In the first experiment, the purpose is to track a person in the *"AVSS_AB_Hard_Divx.avi"* video from the i-LIDS bag and vehicle detection challenge dataset [35], while, in the second experiment, we aim to track a car in the video *"AVSS_PV_Easy_Divx.avi"* from the i-LIDS bag and vehicle detection challenge dataset [35]. The tracking results for the proposed CH-LSK tracker and the PF tracker are illustrated in Figures 1 and 2 for the first and second experiment, respectively. We note that the FT tracker does not take into account changes in scale. Therefore, it was not used in these experiments. We can observe in Figure 1 that the CH-LSK tracker is successful in tracking the change in the object image size, as opposed to PF tracker, which keeps an almost constant size for the tracked object. On the other hand, both trackers have a similar performance in keeping track of the decreasing size of the car in the second experiment (Figure 2).

b) Case study 3: In this experiment we test the performance of the proposed algorithm in the video *"camera8.avi"* of the PETS2001 dataset [36], which depicts a car moving in a circular trajectory in an omnidirectional camera. The experimental results are shown in Figure 3 for the proposed CH-LSK tracker and the PF tracker. Again, the FT tracker is not used, as it does not handle rotational motion. The proposed tracker and the PF tracker have the same initialization. The PF tracker loses the object very quickly, while the CH-LSK tracker follows better both the rotation and scale changes of the car. This means that the proposed tracker is more robust in rotation changes.

c) Case studies 4 and 5: This case study deals with the problem of object tracking in a video with partial occlusion and small scale variations. We conducted two experiments. In the first experiment, the video used is *"OneStopMoveEnter1cor.avi"* from the CAVIAR data set [37]. The object of interest (a man) is partially occluded and his ROI size increases, as he moves towards the camera. The tracking results are depicted in Figure 4. The PF tracker, in the beginning of the video, tracks the full body of the man, but, as tracking evolves, the tracking area gets smaller resulting, after 160 frames, to track only man's torso. The FT tracker is able to handle partial occlusion and tracks the full body of the man successfully. The CH-LSK tracker tracks successfully only man's torso, as the bounding box contains a significant amount of background area, which affects tracking performance.

In the second experiment, the object of interest is the man in the *"WalkByShop1cor.avi"* video of the CAVIAR dataset [37]. In this video, more than 75% of the object area is occluded. The tracking results are shown in Figure 5. PF tracker stops tracking the man, when he walks behind the first person in the foreground (from the right). FT tracker is able to handle the first occlusion but, due to the fact that it cannot follow the person's change in scale, it stops tracking the man when he walks behind the second person in the foreground (from the right). On the other hand, the proposed tracker is able to track the man throughout the video.

d) Case study 6: In this case study, we test the performance of the proposed CH-LSK tracker in a video with strong changes in illumination conditions. More precisely, we track the face of a person, who moves in a room with half of the lights switched on and which are switched off after a while. Snapshots of the tracking results are depicted in Figure 6. We notice that the proposed CH-LSK tracker is robust to illumination variations, as it tracks the person's face, either in the case where the illumination change is gradual, i.e., when the person moves slowly towards the lit part of the room, or in the case where the illumination change is abrupt, i.e., when the lights are switched off. The PF tracker has similar behavior to the CH-LSK tracker, while the FT tracker is not able to handle the gradual illumination change: when the person walks in the lit part of the room it drifts to the person's t-shirt, which has more similar color to the person's face in the previous frames.

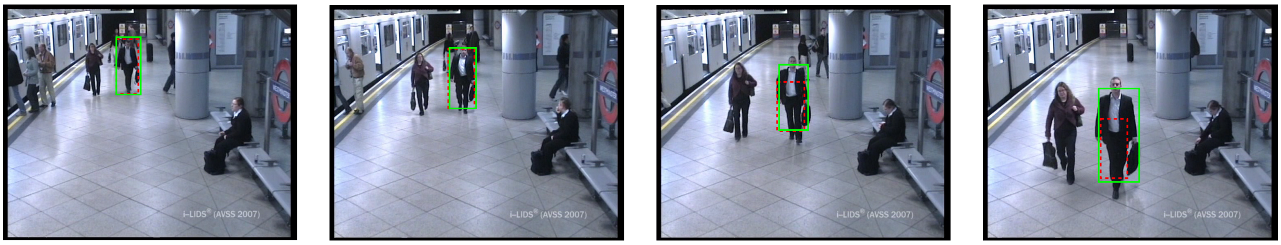
Visual object tracking can be employed in human activity recognition systems such as eating and drinking, by analyzing the trajectories of the employed auxiliary utensils, e.g. glass, spoons, forks, as shown in Figure 7. A drinking activity can be recognized by the trajectory of the glass or by the distance between the glass and the face, as shown in Figures 7-9. In eating activity recognition, the tracked object can be other kitchen utensils (e.g. fork, knife, spoon) or the bare human hands. In the following experiments, we test the performance of the algorithm in tracking objects which can be used in this framework. The test videos were recorded in AIIA laboratory and are included in the MOBISERV-AIIA eating and drinking activity database, which is employed in a nutrition support system designed to prevent dehydration and underfeeding of patients suffering from dementia.

e) Case studies 7-9: In these case studies, we compare the performance of the three trackers when tracking a glass or hands during eating/drinking activity. The video *"drink_left.avi"* depicts a person, when he takes one sip from the glass. The experimental results are shown in Figure 7. The PF tracker cannot keep up with the orientation change of the glass and loses track of it during sipping. The FT tracker loses track of the glass when moving the glass up/down between the table and the mouth, but coincidentally finds the object when it is set back on the table, because its final position is very close to its original one. The CH-LSK tracker is successful in

TABLE I

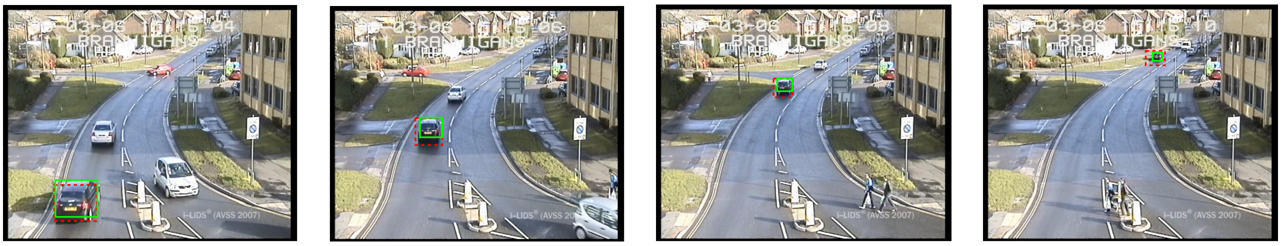
Description of the videos used in case studies 1-9.

case study	length (in frames)	$Q_1 \times Q_2$	medium complexity	indoor/outdoor	illumination changes	object speed	object orientation
1	149	15 × 35	simple	indoor	no	constant	constant
2	190	33 × 31	simple	outdoor	no	constant	constant
3	116	20 × 42	moderate	outdoor	yes	constant	constant
4	298	25 × 37	high	indoor	no	constant	constant
5	148	15 × 32	high	indoor	no	constant	constant
6	431	25 × 34	moderate	indoor	yes	varying	varying
7	77	30 × 29	simple	indoor	no	varying	varying
8	384	30 × 37	simple	indoor	no	varying	varying
9	94	31 × 31	moderate	indoor	no	varying	varying
10	218	28 × 28	high	indoor	no	varying	varying



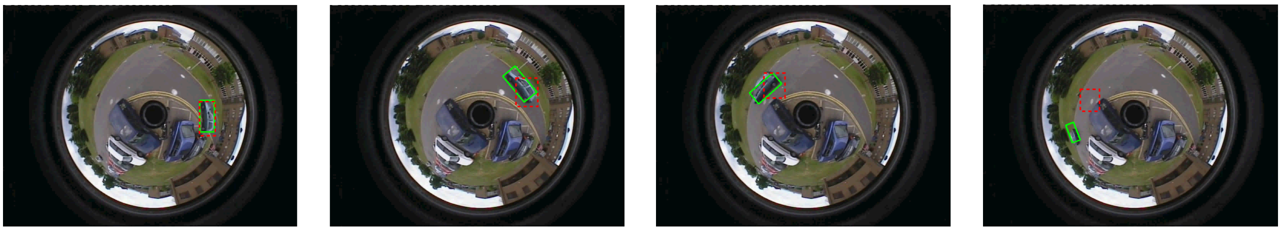
1st frame 45th frame 87th frame 130th frame

Fig. 1. Tracking results of an object having increasing size. Solid line: results of the proposed CH-LSK tracker. Dashed line: results of the particle filter tracker.



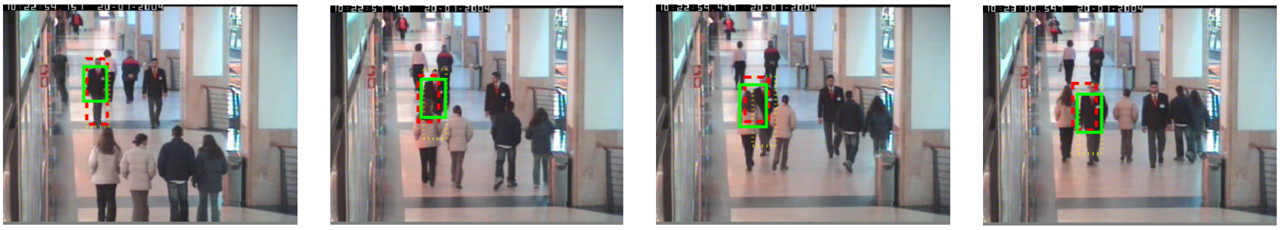
1st frame 43rd frame 95th frame 166th frame

Fig. 2. Tracking results of an object having decreasing size. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker.



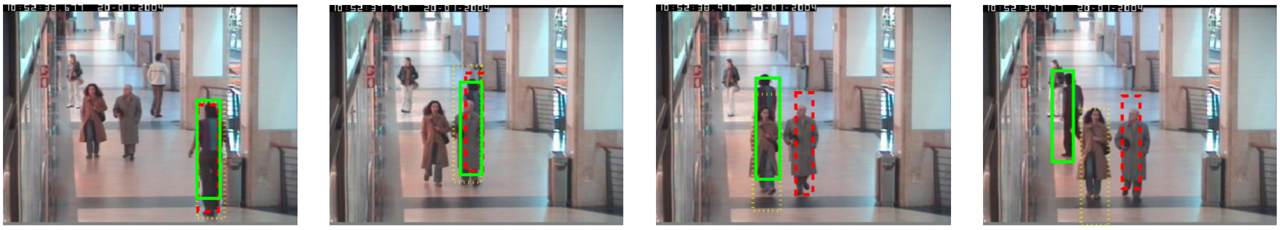
1st frame 25th frame 81st frame 115th frame

Fig. 3. Tracking results in video depicting object rotation. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker.



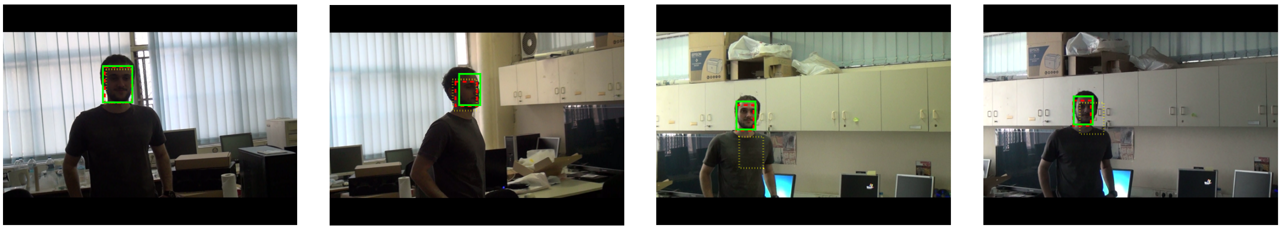
1st frame 91st frame 133rd frame 161st frame

Fig. 4. Tracking results of an object with partial occlusion and small size variations. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker. Dotted bounding box: results of the fragments-based tracker.



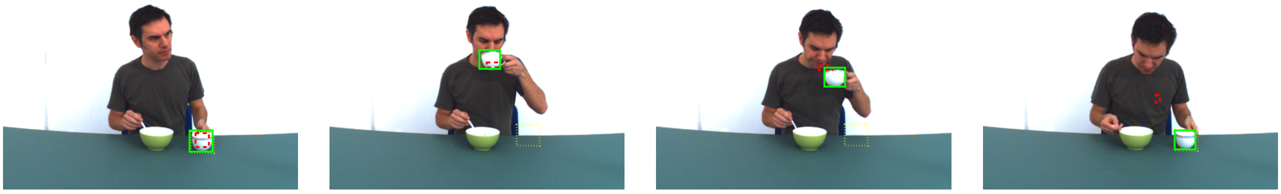
1^{st} frame 103^{rd} frame 131^{st} frame 145^{th} frame

Fig. 5. Tracking results of an object with over 75% partial occlusion. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker. Dotted bounding box: results of the fragments-based tracker.



1^{st} frame 135^{th} frame 263^{rd} frame 285^{th} frame

Fig. 6. Tracking results of an object with strong changes in the illumination conditions. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker. Dotted bounding box: results of the fragments-based tracker.



1^{st} frame 19^{th} frame 52^{nd} frame 76^{th} frame

Fig. 7. Glass tracking during drinking activity. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker. Dotted bounding box: results of the fragments-based tracker.

tracking the glass throughout the duration of the video.

Furthermore, in this experiment, we test the performance of the three trackers in tracking human hands during eating. In the video *"hands.avi"*, the person cuts food with a knife and then eats with a fork. Generally, hand tracking is a difficult task, because hands are articulated objects and they constantly change shape. Figure 8 shows the results of the three trackers. The PF tracker and the FT tracker keep tracking the right hand, but stop tracking the left hand, which performs more complicated movements. The proposed tracker handles successfully the movement of both hands.

In case study 9, we test the performance of the tracker in tracking a glass in an activity which is not drinking. In the video *"glass.avi"* the person enters the scene and sets the glass on the table. The glass of interest changes size and rotation and it is partially occluded by the hands. The experimental results are shown in Figure 9. We notice that all trackers are successful in tracking part of the glass in the whole duration of the video. However, only the CH-LSK tracker is able to track the change in the object size.

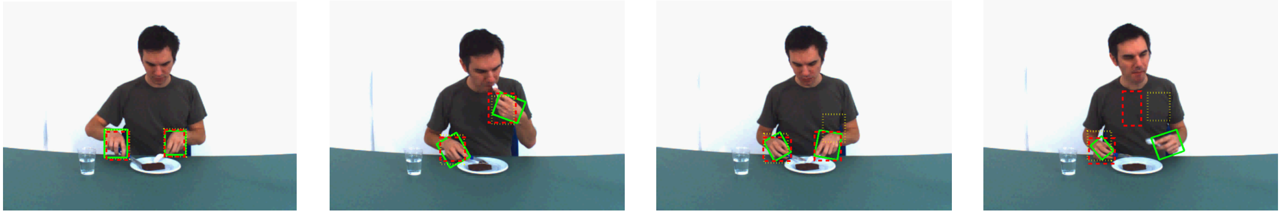
f) Case study 10: In this case study, we test the performance of the algorithm in a complex scenario for face tracking. In the video *"face.avi"*, a face constantly changes orientation and the hands occlude part of the face. The results are shown in Figure 10. In the entire duration of the video, the CH-LSK tracker tracks better the facial area, than either the PF tracker or the FT tracker, which drifts upwards when the person lowers the head and stops tracking the head, when it is shifted to a profile view.

C. Quantitative evaluation

A further quantitative evaluation of the proposed CH-LSK tracker performance and comparison with the other two state of the art trackers is performed through the Frame Detection Accuracy (FDA) measure [38], which calculates the overlap area between the ground truth object G and the detected object D at a given video frame t:

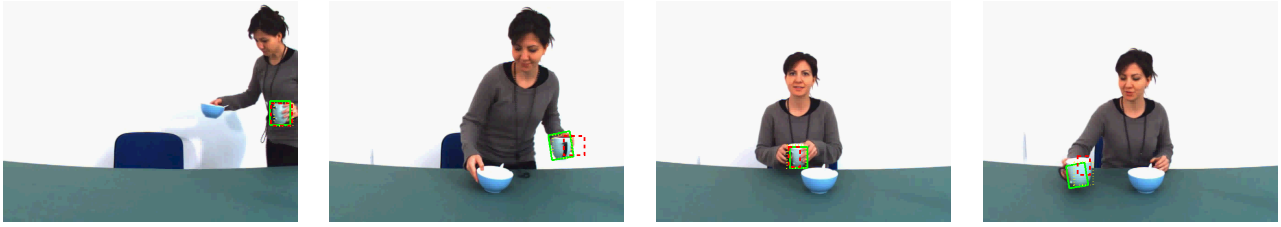
$$FDA(t) = \frac{1}{N_t} \sum_{i=1}^{N_t} \frac{G_i(t) \cap D_i(t)}{G_i(t) \cup D_i(t)}, \tag{13}$$

where N_t is the number of objects in frame t. It takes values in the range from 0 (when the object is lost) to 1 (when tracking is perfect). The ground truth was annotated manually and was defined as the largest bounding box which contains all the visible parts of the object. This rule was also applied in the case of $2D$ object rotation. Figure 11 illustrates the FDA of the LSK, PF and FT trackers, for the videos in the case studies



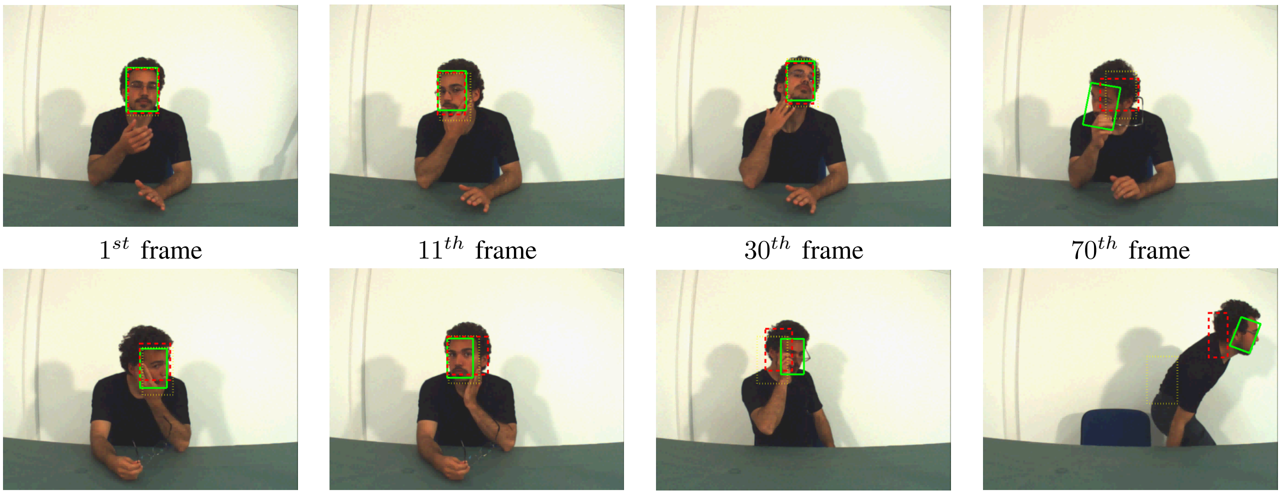
1^{st} frame 73^{rd} frame 130^{th} frame 166^{th} frame

Fig. 8. Hand tracking during eating activity. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker. Dotted bounding box: results of the fragments-based tracker.



1^{st} frame 22^{nd} frame 57^{th} frame 75^{th} frame

Fig. 9. Glass tracking in a table top activity. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker. Dotted bounding box: results of the fragments-based tracker.

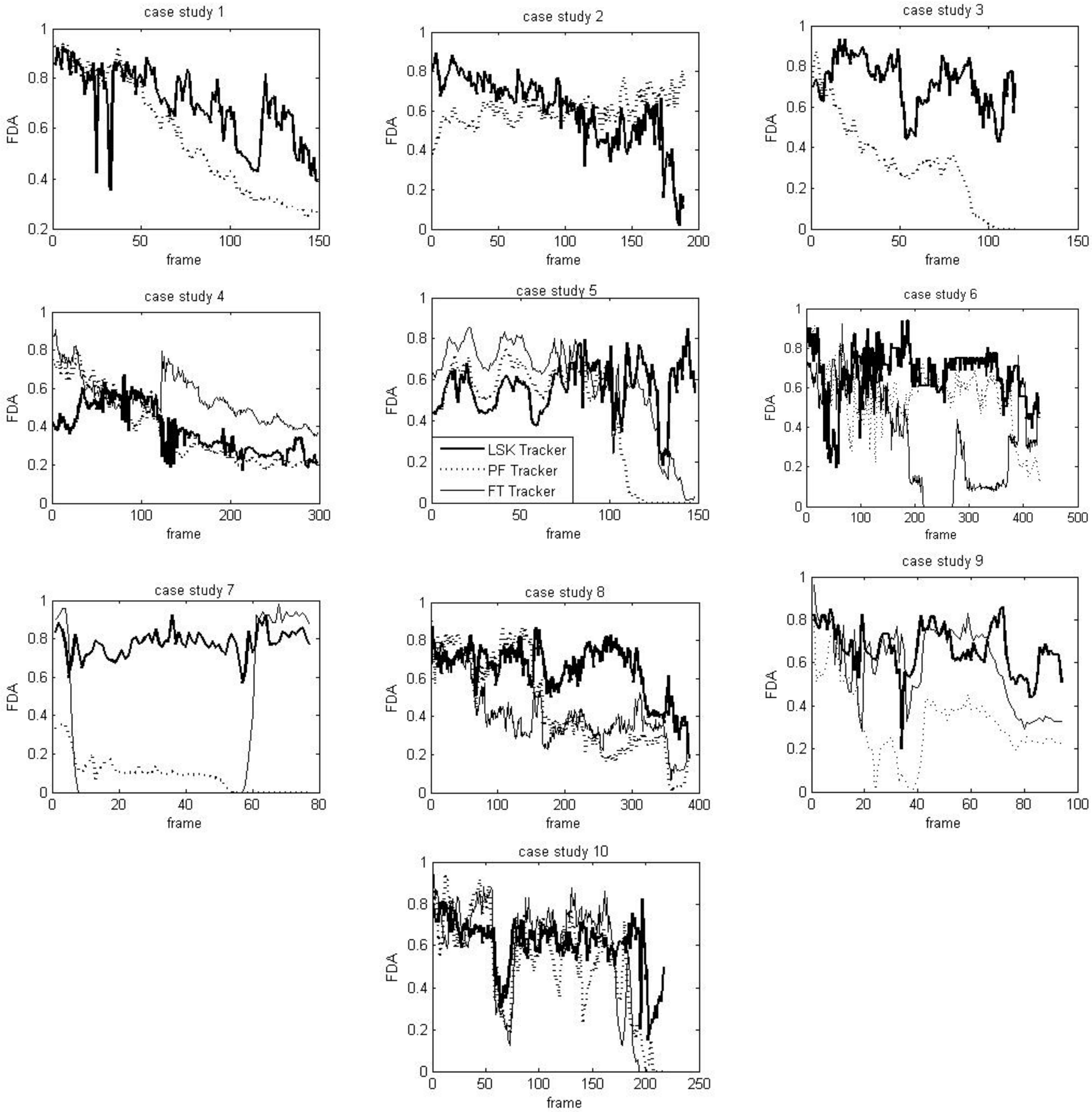


1^{st} frame 11^{th} frame 30^{th} frame 70^{th} frame

120^{th} frame 154^{th} frame 188^{th} frame 216^{th} frame

Fig. 10. Face tracking results under complex movement and occlusion. Solid bounding box: results of the proposed CH-LSK tracker. Dashed bounding box: results of the particle filter tracker. Dotted bounding box: results of the fragments-based tracker.

1-10. We notice that, in case studies 1 and 3, the CH-LSK tracker follows better the object projected size increase and in-plane rotation, than the PF tracker. In case study 2, the FDA of the CH-LSK tracker slightly drops with the decrease of the object size. A larger drop of the FDA is observed, when the object size becomes smaller than $Q_1 \times Q_2$, because the object details are not preserved. On the other hand, the PF tracker is not affected by the strong decrease of the object size, due to its more accurate prediction model. In case studies 4 and 5, the FDA of the CH-LSK tracker is lower than that of the PF and the FT trackers, because it is initialized in a smaller region, so that it does not contain a significant amount of background. Moreover, in case study 5, the CH-LSK tracker is the only one which keeps track of the man during both partial occlusion instances. The PF tracker drifts, when the man walks behind the second man and the FT tracker drifts, when the man walks behind the woman. In case study 6, the CH-LSK and the PF trackers track the object successfully throughout the entire video duration. However, the FDA of the CH-LSK tracker is better than that of the PF tracker in approximately 85% of the video frames. The FT tracker loses track of the object, when the object moves towards the illumination source and finds the object again, when the illumination is switched off. In case study 7, the CH-LSK tracker is the only one which keeps track of the glass throughout the drinking activity duration. The FDA of the PF tracker decreases until it drifts at video frame 55, due to its inability to handle the change of the object speed. The FT tracker drifts the moment the glass is lifted off the table, because the tracker mistakes the line drawn on the glass with the border of the table. In case study 8, the FDA of the CH-LSK tracker is constantly better than the FDA of the PF and FT trackers, due to their inability to track the



case study 1

case study 2

case study 3

case study 4

case study 5

case study 6

case study 7

case study 8

case study 9

case study 10

Fig. 11. Frame Detection Accuracy of the proposed CH-LSK tracker (thick continuous line), the particle filter tracker (dotted line) and the fragments-based tracker (thin continuous line) for the videos in case studies 1-10.

constant changes in motion speed and orientation of the left hand. In case study 9, all three trackers track the change in object appearance. However, the CH-LSK tracker succeeds in tracking a larger object area. Finally, in case study 10, the PF and FT trackers have similar FDA to the CH-LSK tracker in the first 190 frames and drift when the face turns to profile view. On the other hand, the CH-LSK tracker handles better the change in object appearance.

The Average Tracking Accuracy (ATA) is calculated by averaging the FDA over all video frames:

$$ATA = \frac{1}{N} \sum_{t=1}^{N} FDA(t), \tag{14}$$

where N is the total number of frames in the video. The ATA of the three trackers is shown in the first three columns of Table II. It is clear that the proposed tracker has better average tracking accuracy than the other two trackers in 7 out of 10 videos. In case study 2, the racking performance of the PF tracker is 0.5% better than the CH-LSK tracker therefore we can consider that they have equal accuracies. In case studies 4 and 5, the CH-LSK tracker has lower ATA than the FT tracker due to the smaller initialization area of the object. However, in case study 5, CH-LSK tracker is the only one that doesn't drift. The Overall Tracking Accuracy (OTA) is defined as the weighted sum of the respective ATAs in each video:

$$OTA = \frac{1}{N_T} \sum_{i=1}^{n} N_i ATA_i, \qquad (15)$$

where n is the number of videos, N_i is the number of frames in the i^{th} video and $N_T = \sum_{i=1}^{n} N_i$ is the total number of frames. Table II shows that the CH-LSK tracker achieves an OTA of 0.6088, which is approximately 32% better than the OTA of the PF tracker and the FT tracker, respectively.

In order to examine the significance of the LSK features to the proposed tracking framework, we re-performed the experiments in case studies 1-10, without taking into account color-histogram similarity information, i.e., by setting the threshold $\tau_t = 0$ at each video frame. The ATA is shown in the fifth column of Table II. We notice that the use of the color-histogram information increases the ATA in all case studies. The overall tracking performance increases by 27%. Next, we examine the case of using the RGB color space both for the LSK feature and color-histogram extraction. The ATA is shown in the sixth column of Table II. We notice that in all case studies the ATA of the RGB+La*b* CH-LSK tracker is better than the ATA of the RGB-only CH-LSK tracker. This result is in agreement with [26]. Similarly, we examine the case of extracting the color-histograms in the La^*b^* color space instead of the RGB color space (La*b*-only CH-LSK tracker). The tracking accuracy is shown in the seventh column of Table II. We notice that the tracking accuracy of the La^*b^* tracker is lower than the accuracy of the CH-LSK tracker in 7 out of 10 case studies. This is because the RGB color space has better discriminating ability than the La^*b^* color space. Next, we examine the tracking performance, if we replace the LSK descriptors with the Locally Adaptive Regression Kernel (LARK) descriptors [39], which have been employed successfully for face verification [39]. The ATA for the videos in case studies 1-10 are shown in the eighth column of Table II. We notice that LARK descriptors (with CH) achieve better ATA in only 1 out of 10 videos. Therefore, LSK descriptors are more suitable for tracking than LARK descriptors. Finally, we compare the performance of using a linear Kalman filter to using more accurate particle filters. The results are shown in the ninth column of Table II. We notice that the use of particle filters increases the ATA only in 3 out of 10 videos, where the object performs smooth movement. By observing the ATA of the eight trackers of Table II we notice that the proposed CH-LSK tracker, which employs a simple Kalman filter, achieves the highest ATA in 4 out of 10 videos and the highest or second-highest ATA in 8 out of 10 videos. Moreover, the proposed CH-LSK tracker achieves the highest OTA, which is 5.5% better than the OTA of the second best CH-LSK+PF tracker, which employs particle filters.

In the last experiment, we test the significance of the parameter λ (the weight of the LSK similarity of the search region patches to the stored object instance) in the tracking performance. The ATA of the CH-LSK tracker, when parameter λ takes values from 0 to 1 with a step of 0.1 is shown in Table III for the video of case study 6. From Table III we notice that the ATA of the CH-LSK tracker is optimal when λ takes values from 0.4 to 0.7, which is a rather wide range. When the value of λ drops under 0.4, i.e., when more weight is assigned to the object in the previous frame, then the algorithm cannot "understand" when the object appearance changes or when it drifts, hence, the ATA is very low. Likewise, when we increase the weight on the stored object instance (parameter λ takes values greater than 0.7), then the algorithm searches for objects similar to the last stored object instance, even when the object appearance changes significantly, since the last update of the object model. Therefore, the tracking performance drops again.

TABLE III
ATA OF THE CH-LSK TRACKER FOR VARIOUS VALUES OF λ FOR THE VIDEO IN CASE STUDY 8.

λ	ATA	λ	ATA	λ	ATA	λ	ATA
0.0	0.2275	0.3	0.4343	0.6	0.6611	0.9	0.4893
0.1	0.4400	0.4	0.6164	0.7	0.6233	1.0	0.4195
0.2	0.3419	0.5	0.6627	0.8	0.5098		

D. Computational complexity

In this subsection we will provide an analysis on the tracking computational complexity. Given $R_1 \times R_2$ the search region dimensions, the proposed tracker requires $3M^2R_1R_2$ computations of equation (4), which contains 6 multiplications. For taking the decision on the position of the object the algorithm requires $3(R_1 - Q_1 + 1)(R_2 - Q_2 + 1)/d^2$ computations of color-histogram cosine similarities and less than $3(R_1 - Q_1 + 1)(R_2 - Q_2 + 1)/d^2$ computations of LSK cosine similarities. Therefore, the computational complexity of the proposed tracker is upper bounded by $\mathcal{O}(18M^2R_1R_2 + 3(256 + Q_1Q_2M^2)(R_1 - Q_1 + 1)(R_2 - Q_2 + 1)/d^2)$ multiplications.

The tracking speed of the proposed CH-LSK tracker is comparable to the speed of the FT tracker which, unlike CH-LSK tracker, does not take into account the changes of the object scale and in plane rotation. The algorithm speed can be increased if we employ particle filters instead of exhaustive search of the object and if we extract the LSK features in the gray-scale frame, reaching the speed on the PF tracker, which performs near real-time tracking. The algorithm speed can be improved greatly and even achieve real time performance (30fps), if accelerated hardware is used, e.g. GPU, since both the LSK feature extraction, as well as the calculation of the color-histograms and LSKs cosine similarities of the search region patches to the object instances can be executed in parallel.

IV. Conclusion

In this paper, we have proposed a tracking scheme for visual object tracking with online learning of the object model. The tracker extracts a representation of the target object and the video frame based on Local Steering Kernels and color histogram at video frame $t - 1$ and tries to find its location in the frame t, which best suits the object. Each significant change in the object appearance, due to an affine transformation or view change is stored in a stack, representing the target object model. The visual resemblance is determined

TABLE II

ATA AND OTA OF THE CH-LSK, PF AND FT TRACKERS AND OF VARIANCES OF THE CH-LSK TRACKER FOR THE VIDEOS IN CASE STUDIES 1-10.

ATA	CH-LSK tracker	PF tracker	CH tracker	LSK tracker	RGB-only CH-LSK tracker	La^*b^*-only CH-LSK tracker	CH-LARK tracker	CH-LSK+PF tracker
case study 1	*0.6928*	0.5678	-	0.5541	0.6817	0.4885	0.5182	**0.7755**
case study 2	0.5997	0.6029	-	0.2944	0.5346	**0.6338**	*0.6325*	0.5864
case study 3	*0.7149*	0.3247	-	0.6402	0.6114	0.6847	0.5700	**0.7211**
case study 4	0.3789	0.3810	**0.5482**	0.2520	0.3695	0.1910	0.3481	*0.4341*
case study 5	*0.5595*	0.4294	**0.5848**	0.4944	0.4231	0.4165	0.4374	0.4850
case study 6	**0.6627**	0.5474	0.3608	*0.6143*	0.5362	0.0737	0.5032	0.6008
case study 7	**0.7745**	0.0885	0.2755	0.6800	0.5821	*0.7496*	0.5242	0.6193
case study 8	**0.6446**	0.4527	0.4068	0.4433	0.5648	0.4721	0.6017	*0.6067*
case study 9	*0.6600*	0.3220	0.5920	0.4985	0.6251	**0.6949**	0.4745	0.5552
case study 10	**0.6000**	0.5085	0.5540	0.4873	0.3641	*0.5652*	0.4741	0.5104
OTA	**0.6088**	0.4616	0.4602	0.4783	0.5120	0.5183	0.5075	*0.5771*

with respect to the detected object in the previous video frame and the last inserted object instance in the object model stack.

Experimental results showed the effectiveness of the proposed method in object tracking under severe changes in appearance, affine transformations and partial occlusion. The algorithm was successful in the common task of tracking people and cars from surveillance cameras. Moreover, the algorithm performance was tested in the more demanding task of tracking objects manipulated by humans in different activities with constant view changes and/or deformations. More specifically, the method was tested in tracking a cup in both a drinking and a non-drinking activity, human hands while eating and a human face under rotation and pose variations and partial occlusion. The performance of the proposed framework was by far superior to that of the competing state-of-the-art trackers. Further analysis of the object trajectories, as well as information about the sequence of the detected affine transformations, can reveal the motion patterns of objects used in human activities and, moreover, it can be employed in an activity recognition framework.

However, the method has certain limitations. First of all, it does not handle the case of full occlusion. When the object is occluded, the tracker continues tracking another object in the background. The case of full occlusion can be handled by setting an LSK similarity threshold, which stops tracking when the object is lost. The produced object model, i.e., the stored object instances, can then be employed for re-initialization of the object, when it reappears in the video. Possible anomalies that may occur in the video, such as fluctuations or camera failure for 1/2 second, can be handled likewise. Furthermore, in some cases of partial occlusion, the tracker loses track of the target object following a background object. This usually happens when there are similar objects in the background. Moreover, the position prediction method (Kalman filter) cannot follow sudden changes in the object direction or speed. A larger search region could resolve this issue, but it would result in rapid decrease of the algorithm speed. Finally, the tracking speed is rather low, due to brute-force search, rendering it inapplicable in real time applications. Tracking speed can be improved by a more accurate estimation of the object position, scale and angle. Such improvements, as well as possible extension of the proposed tracking framework in multi-camera systems and multiple-object tracking are subject to future research.

REFERENCES

[1] J. Wang, G. Bebis, and R. Miller, “Robust video-based surveillance by integrating target detection with tracking,” in *Conference on Computer Vision and Pattern Recognition Workshop OTCBVS. CVPRW ’06.*, june 2006, pp. 137–145.

[2] N. Papanikolopoulos, P. Khosla, and T. Kanade, “Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision,” *IEEE Transactions on Robotics and Automation*, vol. 9, no. 1, pp. 14–35, February 1993.

[3] G. R. Bradski, “Computer Vision Face Tracking For Use in a Perceptual User Interface,” in *IEEE Workshop on Applications of Computer Vision*, 1998, pp. 214–219.

[4] T. Okuma, “A natural featurebased 3d object tracking method for wearable augmented reality,” in *Proc. of Advanced Motion Control (AMC’04)*, 2004, pp. 451–456.

[5] H. Luo, S. Ci, D. Wu, N. Stergiou, and K. Siu, “A remote markerless human gait tracking for e-healthcare based on content-aware wireless multimedia communications,” *IEEE Wireless Communications*, vol. 17, no. 1, pp. 44 –50, February 2010.

[6] A. Cavallaro, O. Steiger, and T. Ebrahimi, “Tracking video objects in cluttered background,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 4, pp. 575 – 584, April 2005.

[7] D. Roller, K.Daniilidis, and H. H. Nagel, “Model-based object tracking in monocular image sequences of road traffic scenes,” *International Journal of Computer Vision*, vol. 10, pp. 257–281, 1993.

[8] C. Yang, R. Duraiswami, and L. Davis, “Efficient mean-shift tracking via a new similarity measure,” in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.*, vol. 1, June 2005, pp. 176 – 183.

[9] 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*, vol. 26, no. 11, pp. 1531 –1536, nov. 2004.

[10] Y. Wang and O. Lee, “Active mesh-a feature seeking and tracking image sequence representation scheme.” *IEEE Transactions on Image Processing*, vol. 3, pp. 610–624, 1994.

[11] L. Fan, M. Riihimaki, and I.Kunttu, “A feature-based object tracking approach for realtime image processing on mobile devices,” in *17th IEEE International Conference on Image Processing (ICIP)*, sept. 2010, pp. 3921 –3924.

[12] L.-Q. Xu and P. Puig, “A hybrid blob- and appearance-based framework for multi-object tracking through complex occlusions,” in *2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance.*, oct. 2005, pp. 73 – 80.

[13] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, “Real-time foreground-background segmentation using codebook model,” *Real-Time Imaging*, vol. 11, pp. 172–185, June 2005.

[14] M. Piccardi, “Background subtraction techniques: a review,” in *IEEE International Conference on Systems, Man and Cybernetics*, vol. 4, oct. 2004, pp. 3099 – 3104.

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

[16] R. Collins, "Mean-shift blob tracking through scale space," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, June 2003, pp. 234–240.

[17] S. Birchfield and S. Rangarajan, "Spatiograms versus histograms for region-based tracking," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition.*, vol. 2, June 2005, pp. 1158 – 1163.

[18] D. Comaniciu and P. Mee, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, 2002.

[19] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2006, pp. 798–805.

[20] J. Jeyakar, R. V. Babu, and K. Ramakrishnan, "Robust object tracking with background-weighted local kernels," *Computer Vision and Image Understanding*, vol. 112, no. 3, pp. 296 – 309, 2008.

[21] C. Yang, R. Duraiswami, and L. Davis, "Fast multiple object tracking via a hierarchical particle filter," in *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005.*, vol. 1, Oct. 2005, pp. 212 – 219 Vol. 1.

[22] S. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Transactions on Image Processing*, vol. 13, pp. 1434–1456, 2004.

[23] A. Doucet, N. D. Freitas, and N. Gordon, Eds., *Sequential Monte Carlo methods in practice*, 2001.

[24] R. Marfil, L. Molina-Tanco, J. Rodrguez, and F. Sandoval, "Real-time object tracking using bounded irregular pyramids," *Pattern Recognition Letters*, vol. 28, no. 9, pp. 985 – 1001, 2007.

[25] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental Learning for Robust Visual Tracking," *International Journal of Computer Vision*, vol. 77, no. 1, pp. 125–141, May 2008.

[26] H. Seo and P. Milanfar, "Training-free, generic object detection using locally adaptive regression kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1688–1704, Sept. 2010.

[27] O. Zoidi, A. Tefas, and I. Pitas, "Object tracking based on local steering kernels for drinking activity recognition," in *33rd International Conference on Information Technology Interfaces (ITI)*, june 2011, pp. 237 –242.

[28] H. Takeda, S. Member, S. Farsiu, P. Milanfar, and S. Member, "Kernel regression for image processing and reconstruction," *IEEE Transactions on Image Processing*, vol. 16, pp. 349–366, 2007.

[29] J. Movellan, "Tutorial on gabor filters," *technical report, MPLab Tutorials*, 2005.

[30] A. Bors and I. Pitas, "Prediction and tracking of moving objects in image sequences," *IEEE Transactions on Image Processing*, vol. 9, pp. 1441–1445, 2000.

[31] H. J. Seo and P. Milanfar, "Static and space-time visual saliency detection by self-resemblance." *Journal of Vision*, vol. 9, no. 12, pp. 1–27, 2009.

[32] G. Welch and G. Bishop, "An introduction to the kalman filter," in *Chapel Hill, Tech. Rep. TR95041*, 2000.

[33] S. Maskell and N. Gordon, "A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, pp. 174–188, 2001.

[34] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 603–619, 2002.

[35] [Online]. Available: http://www.eecs.qmul.ac.uk/~andrea/avss2007$_$d.html

[36] [Online]. Available: ftp://ftp.pets.rdg.ac.uk/pub/PETS2001/

[37] [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/

[38] R. Kasturi, D. Goldgof, P. Soundararajan, V. Manohar, J. Garofolo, R. Bowers, M. Boonstra, V. Korzhova, and J. Zhang, "Framework for performance evaluation of face, text, and vehicle detection and tracking in video: Data, metrics, and protocol," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 319 –336, Feb. 2009.

[39] H. J. Seo and P. Milanfar, "Face verification using the lark representation," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 4, pp. 1275 –1286, dec. 2011.

Olga Zoidi received the B.Sc. in Mathematics and the diploma in Electrical and Computer Engineering in 2004 and 2009, respectively, both from the Aristotle University of Thessaloniki. She is currently a Ph.D. student in the Artificial Intelligence and Information Analysis laboratory at the Department of Informatics of Aristotle University of Thessaloniki. Her research interests include image and video processing, computer vision and pattern recognition.

Anastasios Tefas (M'04) received the B.Sc. in informatics in 1997 and the Ph.D. degree in informatics in 2002, both from the Aristotle University of Thessaloniki, Greece. Since 2008, he has been a Lecturer at the Department of Informatics, Aristotle University of Thessaloniki. From 2006 to 2008, he was an Assistant Professor at the Department of Information Management, Technological Institute of Kavala. From 2003 to 2004, he was a temporary lecturer in the Department of Informatics, University of Thessaloniki. From 1997 to 2002, he was a researcher and teaching assistant in the Department of Informatics, University of Thessaloniki. Dr. Tefas participated in 10 research projects financed by national and European funds. He has co-authored 29 journal papers, 91 papers in international conferences and contributed 7 chapters to edited books in his area of expertise. Over 1600 citations have been recorded to his publications and his H-index is 20 according to Google scholar. His current research interests include computational intelligence, pattern recognition, statistical machine learning, digital signal and image processing and computer vision.

Ioannis Pitas (SM'94–F'07) received the Diploma and Ph.D. degree in Electrical Engineering, both from the Aristotle University of Thessaloniki, Greece. Since 1994, he has been a Professor at the Department of Informatics of the same University. He served as a Visiting Research Associate or Visiting Assistant Professor at several Universities. His current interests are in the areas of intelligent digital media, image/video processing (2D/3D) and human-centered interfaces. He has published over 690 papers, contributed in 39 books in his areas of interest and edited or (co-)authored another 8 books. He has also been an invited speaker and/or member of the program committee of many scientific conferences and workshops. In the past he served as Associate Editor or co-Editor of eight international journals and General or Technical Chair of four international conferences (including ICIP2001). He participated in 65 R&D projects, primarily funded by the European Union and is/was principal investigator/researcher in 40 such projects. He has 14418+ citations (Source Publish and Perish), 4937+ (Scopus) to his work and H-index 58+ (Source Publish and Perish), 35+ (Scopus).