Trajectory-based Representation of Human Actions

Antonios Oikonomopoulos[1], Ioannis Patras[2], Maja Pantic[1], and Nikos Paragios[3]

[1]Imperial College London, 180 Queensgate, SW7 2AZ London, UK
{aoikonom,m.pantic}@imperial.ac.uk
[2]Department of Electronic Engineering, Queen Mary University, London, UK
I.Patras@elec.qmul.ac.uk
[3] Ecole Centrale de Paris, Grande Voie des Vignes, 92 295 Chatenay-Malabry, FRANCE
nikos.paragios@ecp.fr

Abstract. This work addresses the problem of human action recognition by introducing a representation of a human action as a collection of short trajectories that are extracted in areas of the scene with significant amount of visual activity. The trajectories are extracted by an auxiliary particle filtering tracking scheme that is initialized at points that are considered salient both in space and time. The spatiotemporal salient points are detected by measuring the variations in the information content of pixel neighborhoods in space and time. We implement an online background estimation algorithm in order to deal with inadequate localization of the salient points on the moving parts in the scene, and to improve the overall performance of the particle filter tracking scheme. We use a variant of the Longest Common Subsequence algorithm (LCSS) in order to compare different sets of trajectories corresponding to different actions. We use Relevance Vector Machines (RVM) in order to address the classification problem. We propose new kernels for use by the RVM, which are specifically tailored to the proposed representation of short trajectories. The basis of these kernels is the modified LCSS distance of the previous step. We present results on real image sequences from a small database depicting people performing 12 aerobic exercises.

1 Introduction

With an ever-increasing role of computers and other digital devices in our society, one of the main foci of the research in Artificial Intelligence should be on Emerging Human-Machine Systems. A related, crucial issue is that of Human-Machine Interaction (HMI). A long-term goal in HMI research is to approach the naturalness of human-human interaction. This means integrating 'natural' means that humans employ to interact with each other into the HMI designs. With this motivation, automatic speech recognition and synthesis have been the topics of research for decades. Recently, other human interactive modalities such as facial and body gestures have also gained interest as potential modes of HMI.

The analysis of what is present in a scene is an essential issue in the development of natural Human-Machine interfaces. Who is in the scene, what is (s)he doing, and how is (s)he doing it, are essential questions that should be answered if natural interfaces that are non-obtrusive and informative for the user are to be realized. Vision offers the means to achieve this. It also represents an essential link to the creation of systems which are able to adapt to the affective state of their user, leading in this way to an affect-sensitive interaction between the user and the machine. Particularly for ambient intelligence, anticipatory interfaces, and human computing, the key is the ease of use - the ability to unobtrusively sense certain behavioral cues of the users and to adapt automatically to their typical behavioral patterns and the context in which they act [1]. Sensing and interpretation of human behavioral cues play an important role and are extremely relevant for the development of applications in the fields of security (video surveillance and monitoring), natural multimodal interfaces, augmented reality, smart rooms, object-based video compression and driver assistance.

2 Related Work

A major component in human computing research is localization and tracking of the human body, either partly (e.g. gesture analysis) or as a whole. Tracking of the articulated human body is inherently a very difficult problem. Its high degree of freedom (usually 28-60) calls upon sophisticated tracking algorithms that can address the problem of high dimensionality. Furthermore, large appearance changes, occlusion between body parts and the absence of typical appearance due to clothing pose additional problems that need to be dealt with.

The main objective of tracking is to predict the state x_k given all the measurements $z_{1:k}$ up to the current time instant. This translates in the construction of the probability density function $p(x_k|z_{1:k})$. State estimation techniques have been extensively used in order to reach a solution to this problem. Theoretically, the optimal solution in case of Gaussian noise in the measurements is given by the Kalman filter [2], which yields the posterior being also Gaussian. Kalman filters and their variants, like the Extended (EKF) and the Unscented Kalman Filters (UKF) [3], [4], [5] have been extensively used for a variety of tracking applications [6], [7]. However, in nonlinear and non-Gaussian state estimation problems Kalman filters can be significantly off.

To overcome the limitations of Kalman filtering, the classical particle filtering algorithm, or so-called Condensation algorithm was proposed [8], [9]. The main idea behind particle filtering is to maintain a set of possible solutions called particles. Each particle is characterized by its location and its weight, the latter expressing the likelihood of the particle being close to the actual solution. By maintaining a set of solutions instead of a single estimate as is done by Kalman filtering, particle filters are more robust to missing and inaccurate data. The major drawback of the classic Condensation algorithm, however, is that a large amount of particles might be wasted because they are propagated into areas with small likelihood. In order to overcome this problem, a number of variants

to the original algorithm have been proposed, having as a common characteristic the goal of achieving a more optimal allocation of new particles. Since particle weights determine how the particles are being resampled, the likelihood function has an essential influence on the tracking performance [10]. Several attempts have been made in order to adjust the way new particles are assigned, through the use of kernels [11], [12], [13], orientation histograms [14] or special transformations like Mean Shift [15]. In all these approaches, the particles are re-weighted not based on the observation likelihood, but on the density estimate of the particular method used.

Despite the improvement in the tracking performance of the previous methods, the inherent problem of the classic condensation algorithm, that is, the decay of the particle set into a single particle is not sufficiently addressed, since it is not guaranteed that the particles will retain their diversity at any time instance. In order to effectively deal with this issue, the Auxiliary Particle Filtering (APF) algorithm was proposed by Pitt and Shephard [16]. The main difference of the latter with the classic condensation algorithm is as follows: particles are initially thrown and their likelihood is evaluated. Subsequently, the algorithm goes one step back, and new particles are created at the positions where the likelihood calculated at the previous step is maximized. Since the introduction of the APF algorithm, a number of variants have been proposed in order to address different issues. In [17] a modified APF tracking scheme is proposed for the tracking of deformable facial features, like mouth and eye corners. The method uses an invariant color distance that incorporates a shape deformation term as an observation model in order to deal with the deformations of the face. In order to take into account spatial constraints between tracked points, the particle filter with factorized likelihoods is proposed in [18]. In this approach, the spatial constraints between different facial features are pre-learned and the proposed scheme tracks constellations of points instead of a single point, by taking into account these constraints.

Particle filters belong to the family of template tracking algorithms. An object is tracked through a video sequence by extracting an example image of the object, or a template, in the first frame. Subsequently, the tracking is performed by finding the region in the successive frames that best matches the extracted template. The underlying assumption behind template tracking is that the object will not significantly change its appearance throughout the duration of the video. This assumption, however, is not realistic, since an object can undergo several rotations, deformations or partial occlusions, making the template no longer an accurate model of the appearance of the object. A simple but rather naive solution to this problem is to update the template at every frame with a new template corresponding to the tracked position of the object. This approach, however, leads to error accumulation, as small errors are constantly introduced in the location of the template. As a result, the template eventually drifts away from the object and in the most cases gets stuck on the static background of the scene. A compromising solution between these two extremes is to maintain a significant portion of the initial template (e.g. $> 90\%$) and slightly change it

over time, a process often called exponential forgetting. Although this solution offers a somewhat more realistic tracking, by allowing the template to adapt, it does not avoid error accumulation, and there is still a high probability that the template will eventually drift away from the object.

Matthews *et al* specifically address the drift problem in [19]. The tracking template is updated at every frame, while maintaining the initial template of the first frame. To eliminate drift, the new template is aligned every time to the initial one using a gradient descent rule. This strategy, however, is most suitable for tracking rigid objects (e.g. cars). For objects whose appearance changes over time, the authors adopt an approach of template tracking with Active Appearance Models (AAM). The appearance model and the template are updated at every time instance, leading to more robust tracking algorithm. A similar framework is presented in [20], where a set of adaptive appearance models are used for motion-based tracking. The appearance model used consists of three components. The stable component (S) is used to capture the behavior of temporally stable and slowly varying image observations, the data outlier or 'lost' component (L) is used to capture data outliers due to failures in tracking, occlusion or noise and finally the 'wandering' component (W) is used to model sudden changes in the appearance of the object. The parameters of the model are adjusted online via EM and the system is tested in tracking scenarios where a high degree of partial object occlusion occurs. Finally, in [21] a Support Vector Machine (SVM) is used in order to provide an initial guess for an object position in the first frame. The position of the initial guess in subsequently refined so that a local maximum of the SVM score is achieved. The whole framework is called Support Vector Tracking (SVT) and is implemented in moving vehicle tracking scenarios.

In contrast to rigid objects, tracking of articulated objects is inherently a much more difficult problem, mainly due to the high number of degrees of freedom that are involved. Accurate human body tracking, in particular, is an extremely important aspect for human computing applications. A possible strategy for estimating the configuration of articulated objects is sequential search, in which a number of parameters are initially estimated and subsequently others, assuming that the first estimation was correct. For instance, Gavrila and Davis in [22] first locate the torso of the human body and then use this information in order to initialize a search for the limbs. This approach, however, only works for specific views and is very sensitive to self-occlusion. A similar approach is presented in [23], where a particle filtering framework is used for the purposes of hand tracking. For the same purpose, Cipolla *et al* [24] propose a view-based hierarchical probabilistic tracking framework that can deal with changes in view and self occlusions. The system uses edge and color cues in order to estimate the likelihood function of the hand position and configuration and subsequently a Bayesian filtering framework that performs the tracking. In [25] a particle filtering approach is adopted for articulated hand tracking. The tracker is guided by attractors, pre-collected training samples of possible hand configurations whos observations are known, while the whole process is mod-

elled by a Dynamic Bayesian Network. A Bayesian Network is also adopted in [26] in order to model the existing constraints between the different parts of the human body. These constraints are learned using Gaussian Mixture Models (GMM) and training is done using ground truth motion-capture frames of walking data. Observations are based on multi-scale edge and ridge filters while the whole process is assisted with a pooled background model derived by the set of training images. In [27] a Dynamic Markov Network is utilized instead to model the relations between body parts and tracking is done using an sequential Monte Carlo algorithm. A similar approach is presented in [28], where an elastic model is used to represent relations and constraints between the limbs and a Nonparametric Belief Propagation (NBP) algorithm for the purpose of tracking.

Articulated object tracking, and particularly human body tracking suffer from dimensionality issues, an inherent problem whenever there is a large number of degrees of freedom. This fact makes the use of tracking algorithms like particle filters rather intractable. The reason for this is that a very large number of particles is required in order to represent the posterior function in a sufficient way, making this kind of tracking alorithms slow and computationally expensive. The problem becomes even more prominent whenever real-time performance is required, such as in monitoring applications, virtual trainers or augmented reality applications. In order to deal with this issue, a number of different techniques have been proposed, either by constraining the configuration space [22] or by restricting the range of the movements of the subject [29]. These approaches, however, greatly reduce the generality of the implemented trackers, making them impractical in real applications. Eigenspace decomposition and principal component analysis offer an intresting alternative for dimensionality reduction. Black and Jepson [30] proposed an algorithm for matching and tracking rigid and articulated objects in time called eigentracking. As a template, the authors construct a model of a known object by using only a small set of views. Using eigenspace decomposition, the view of an object is found and the transformation that maps the object to its model is recovered. Subsequently, this transformation is refined by an optimization technique at any time instant, effectively tracking the object. A Pricipal Component Analysis (PCA) approach is adopted in [31] in order to reduce the dimensionality of the search space in an articulatied hand tracking scenario. Input data are captured with the use of a data glove and PCA is applied initially to reduce the dimensionality of the measurements. Subsequently, a process called Independent Component Analysis (ICA) is implemented in order to extract a number of intrinsic features that correspond to individual finger motions. In [32], a modified particle filtering approach is used in order to reduce the complexity of human body tracking. The latter is modeled by a kinematic chain of 29 degrees of freedom. The main characteristic of the utilized tracker is its ability to avoid local maxima in the tracking by incorporating a search based on simmulated annealing, and thus called annealed particle filter. Apart from dimensionality reduction techniques, several researchers have attempted to modify the way classical tracking algorithms work in order to achieve computational efficiency and real-time tracking performance. A simple example are

the earlier mentioned kernel-based particle filters [11], [12], [13], or particle filters that use special transformations, as in [14], [15]. These methods attempt to limit the number of required particles for efficient tracking, effectively reducing the computational complexity of their algorithms.

Apart from articulated and single object tracking, a large variety of algorithms have been proposed that deal with the problem of multi-target tracking. The inherent problem in these scenarios is how to distinguish and track specific objects in the same scene without any confusion between them and between other moving objects. Possible applications include tracking in crowd and office environments [33], [34], sports [35] and pedestrian tracking [36], [37]. Furthermore, tracking of a specific object in a non static environment (e.g. single pedestrians on the street) also poses several difficulties, as the tracker must not be confused by other moving objects that may occlude or interfere with the tracked object. In [38] an ensemble tracking algorithm is presented for single object tracking in pedestrian or office environments. Tracking is treated as a binary classification problem, where an ensemble of weak classifiers is trained online in order to distinguise between the background and the object. The weak classifiers are selected using AdaBoost and are used to create a confidence map concerning the position of the object in question. Subsequently, Mean Shift is used to accurately locate the object. A similar approach is presented in [39], where tracking is dealt as a developing distinction between the foreground and the background. The background and the foreground are modeled as sets of texture patterns. During tracking, the proposed method uses a set of discriminant functions, trained using Linear Discriminant Analysis (LDA), in order to distinguish between foreground and background patterns.

A wide variety of hand and body tracking methods edge and color cues [14], [12] or some form of markers [40],[41] in order to assist the initialization and the overall operation of the tracking process. In order to avoid the use of markers, an interesting alternative could be the use of interesting points for tracker initialization. According to Haralick and Shapiro [42] an interesting point is a) distinguishable from its neighbors and b) its position is invariant with respect to the expected geometric transformation and to radiometric distortions. Gilles introduces the notion of saliency in terms of local signal complexity or unpredictability in [43]. Kadir and Brady [44] extend the original Gilles algorithm and estimate the information content of pixels in circular neighborhoods at different scales in terms of the entropy. Local extremes of changes in the entropy across scales are detected and the saliency of each point at a certain scale is defined in terms of the entropy and its rate of change at the scale in question.

In this work, we propose a human action recognition algorithm that is based on the detection and tracking of spatiotemporal features in given image sequences. We do this by extending in the temporal direction the salient feature detector developed in [44]. The detected salient points correspond to peaks in activity variation such as the edges of a moving object. Similar to [45], we treat the action as three-dimensional events, by detecting the salient points in the space-time domain. Contrary to [12], [32], [26] and [27] that use models to rep-

resent the human body, we propose an entirely unsupervised method based on the detected salient features in order to represent the moving parts of the body. In this sense, the concept of our method resembles the one in [46], where detection is done without prior knowledge of the types of events, their models, or their temporal extent. Like in [44], we automatically detect the scales at which the entropy achieves local maxima and form spatiotemporal salient regions by clustering spatiotemporal points with similar location and scale. We derive a suitable distance measure between sets of salient regions, which is based on the Chamfer distance, and we optimize this measure with respect to a number of temporal and scaling parameters. In this way we achieve invariance against scaling and we eliminate the temporal differences between the representations. We extend our previous work on salient points presented at [47] by using the detected salient regions in order to initialize a tracking scheme based on the auxiliary particle filter, proposed in [16]. Each image sequence is then represented as a set of short trajectories. The spatiotemporal coordinates of the points that consist the extracted trajectories are appropriately transformed according to the parameters that were estimated in the Chamfer distance optimization step. We use the adaptive background estimation algorithm presented in [48] in order to model the background in the available sequences and to improve the overall quality of the implemented tracking scheme. We use a variant of the Longest Common Subsequence algorithm (LCSS) that was proposed in [49],[50] in order to compare different sets of trajectories. We use Relevance Vector Machines [51] in order to address the classification problem. We propose new kernels for use by the RVM, which are specifically tailored to the proposed short trajectory representation. The basis of these kernels is the modified LCSS distance of the previous step. The novelty of the proposed method lies in the unsupervised nature of representation of the actions. Since we don't use any model, the method can be easily extended and used for a variety of different actions, ranging from full-body actions to single gestures.

We test the proposed method using real image sequences of subjects performing several aerobic exercises. Possible applications lie in the area of e-health, where the development of non-stationary, non-intrusive, non-invasive monitoring inside and outside the clinical environment is essential, due to demanding patients, aging population and rising costs. The method can be realized as an adaptive system that will be able to monitor and assess the correctness of the performed exercise, and will provide an appropriate alternative (senior) fitness plan, assisting in this way nurses,physical therapists and family members. The system can also be configured for use at home, to accommodate elderly but otherwise healthy patients or patients suffering from conditions like rheumatism and chronic pain.

The remainder of the paper is organized as follows: In section 3, the spatiotemporal feature detector used is described, along with the proposed space-time warping technique. In section 4, the auxiliary particle filter that was used is briefly analyzed along with the background estimation model that was utilized. In section 5 the proposed kernel-based recognition method is described. In sec-

tion 6, we present our experimental results, and in section 7, final conclusions are drawn.

3 Spatiotemporal Salient Points

3.1 Spatiotemporal Saliency

Let us denote by $N_c(s, \boldsymbol{v})$ the set of pixels in an image I that belong to a circular neighborhood of radius s, centered at pixel $\boldsymbol{v} = (x, y)$. In [44], in order to detect salient points in static images, Kadir and Brady define a saliency measure $y_D(s, \boldsymbol{v})$ based on measuring changes in the information content of N_c for a set of different circular radiuses (i.e. scales). In order to detect spatiotemporal salient points at peaks of activity variation we extend the Kadir's detector by considering cylindrical spatiotemporal neighborhoods at different spatial radiuses s and temporal depths d. More specifically, let us denote by $N_{cl}(\boldsymbol{s}, \boldsymbol{v})$ the set of pixels in a cylindrical neighborhood of scale $\boldsymbol{s} = (s, d)$ centered at the spatiotemporal point $\boldsymbol{v} = (x, y, t)$ in the given image sequence. At each point $\boldsymbol{v}$ and for each scale $\boldsymbol{s}$ we will define the spatiotemporal saliency $y_D(\boldsymbol{s}, \boldsymbol{v})$ by measuring the changes in the information content within $N_{cl}(\boldsymbol{s}, \boldsymbol{v})$. Since we are interested in activity within an image sequence, we consider as input signal the convolution of the intensity information with a first-order Gaussian derivative filter. Formally, given an image sequence $I_0(x, y, t)$ and a filter G_t, the input signal that we use is defined as:

$$I(x, y, t) = G_t * I_0(x, y, t)\,. \tag{1}$$

For each point $\boldsymbol{v}$ in the image sequence, we calculate the Shannon entropy of the signal histogram in a cylindrical neighborhood $N_s(\boldsymbol{s}, \boldsymbol{v})$ around it. That is,

$$H_D(s, d, \boldsymbol{v}) = -\sum_{q \in D} p(q, s, d, \boldsymbol{v}) \log p(q, s, d, \boldsymbol{v})\,, \tag{2}$$

The set of scales at which the entropy is peaked is given by:

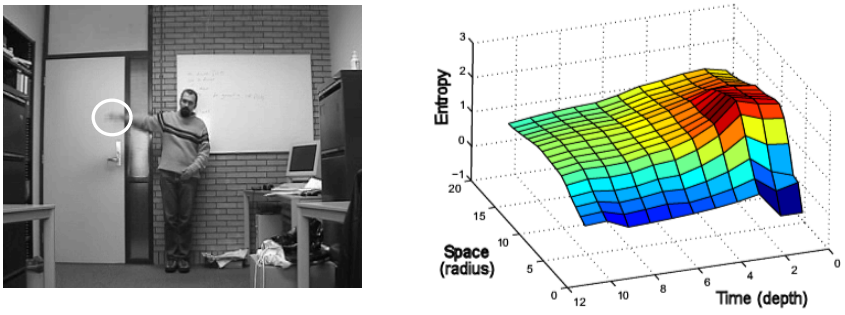
$$\begin{aligned} \hat{S}_p = \{(s, d) : H_D(s-1, d, \boldsymbol{v}) < H_D(s, d, \boldsymbol{v}) > H_D(s+1, d, \boldsymbol{v}) \\ \wedge H_D(s, d-1, \boldsymbol{v}) < H_D(s, d, \boldsymbol{v}) > H_D(s, d+1, \boldsymbol{v})\} \end{aligned} \tag{3}$$

The saliency measure at the candidate scales is given by:

$$y_D(s, d, \boldsymbol{v}) = H_D(s, d, \boldsymbol{v}) W_D(s, d, \boldsymbol{v}), \quad \forall\, (s, d) \in \hat{S}_p\,, \tag{4}$$

The first term of eq. 4 is a measure of the variation in the information content of the signal. The weighting function $W_D(s, \boldsymbol{v})$ is a measure of how prominent the local maximum is at s, and is given by:

$$\begin{aligned} W_D(s, d, \boldsymbol{v}) = \frac{s^2}{2s-1} \sum_{q \in D} |p(q, s, d, \boldsymbol{v}) - p(q, s-1, d, \boldsymbol{v})| \\ + d \sum_{q \in D} |p(q, s, d, \boldsymbol{v}) - p(q, s, d-1, \boldsymbol{v})|, \forall\, (s, d) \in \hat{S}_p\,, \end{aligned} \tag{5}$$



(a) (b)

Fig. 1. (a) Single frame from a sample image sequence where the subject is raising its right hand and (b) the corresponding entropy plot as a function of the spatial radius and temporal depth of all the applied cylindrical neighborhoods. The origin of all the applied cylindrical neighborhoods is the center of the white circle in (a).

where the values in front of each summation in the right part of eq. 5 are normalization factors. When a peak in the entropy for a specific scale is distinct, then the corresponding pixel probability density functions at the neighboring scales will differ substantially, giving a large value to the summations of eq. 5 and thus, to the corresponding weight value assigned. On the contrary, when the peak is smoother, then the summations in eq. 5 will have a smaller value. Let us note that we considered cylindrical neighborhoods for simplicity reasons. However, more complicated shapes, such as elliptical neighborhoods at different orientations and with different axes ratios could be considered.

In Fig. 1(a), a single frame from a sample image sequence is presented, where the subject is raising its right hand. By selecting as origin the center pixel of the drawn white circle, we apply a number of cylindrical neighborhoods of various scales in the sequence and we calculate the corresponding entropy values. The result is shown in Fig. 1(b), where the various entropy values are plotted with respect to the radiuses and depths of the corresponding cylindrical neighborhoods. The scale which corresponds to the distinct peak of the plot is considered candidate salient scale, and is assigned a saliency value, according to eq. 4.

3.2 Salient Regions

The analysis of the previous section leads to a set of candidate spatiotemporal salient points $S = \{(\boldsymbol{s}_i, \boldsymbol{v}_i, y_{D,i})\}$, where $\boldsymbol{v}_i = (x, y, t)$, $\boldsymbol{s}_i$ and $y_{D,i}$ are respectively, the position vector, the scale and the saliency value of the feature point with index i. In order to achieve robustness against noise we follow a similar approach as that in [44] and develop a clustering algorithm, which we apply to the detected salient points. By this we define salient regions instead of salient

points, the location of which should be more stable than the individual salient points, since noise is unlikely to affect all of the points within the region in the same way. The proposed algorithm removes salient points with low saliency and creates clusters that are a) well localized in space, time and scale, b) sufficiently salient and c) sufficiently distant from each other. The steps of the proposed algorithm can be summarized as follows:

1. Derive a new set S_T from S by applying a global threshold T to the saliency of the points that consist S. Thresholding removes salient points with low saliency, that is,
$$S_T = \{(\boldsymbol{s}_i, \boldsymbol{v}_i, y_{D,i}) : y_{D,i} > T\}\,. \tag{6}$$
2. Select the point i in S_T with the highest saliency value and use it as a seed to initialize a salient region R_k. Add nearby points j to the region R_k as long as the intra-cluster variance does not exceed a threshold T_V. That is, as long as
$$\frac{1}{|R_k|}\sum_{j\in R_k} c_j^2 < T_V\,, \tag{7}$$
where R_k is the set of the points in the current region k and c_j is the Euclidean distance of the jth point from the seed point i.
3. If the overall saliency of the region R_k is lower than a saliency threshold T_S,
$$\sum_{j\in R_k} y_{D,j} \leq T_S\,, \tag{8}$$
discard the points in the region back to the initial set of points and continue from step 2 with the next highest salient point. Otherwise, calculate the Euclidean distance of the center of region R_k from the center of salient regions already defined, that is, from salient regions $R_{k'}, k' < k$.
4. If the distance is lower than the average scale of R_k, discard the points in R_k back to the initial set of points, and continue with the next highest salient point. Otherwise, accept R_k as a new cluster and store it as the mean scale and spatial location of the points in it.
5. Form a new set S_T consisting of the remaining salient points, increase the cluster index k and continue from step 2 with the next highest salient point.

By setting the threshold T_V in step 2, we define clusters that have local support and are well localized in space and time. In addition, we want to take the saliency of the points into consideration such that the overall saliency of the region is sufficient. We do this in step 3, by setting a saliency threshold, T_S. Finally, the purpose of step 4 is to accept clusters that are sufficiently distant from each other. To summarize, a new cluster is accepted only if it has sufficient local support, its overall saliency value is above the saliency threshold, and it is sufficiently distant in terms of Euclidean distance from already existing clusters.

3.3 Space-Time Warping

There is a large amount of variability between feature sets due to differences in the execution speed of the corresponding actions. Furthermore, we need to compensate for possible shifting of the representations forward or backward in time,

caused by imprecise segmentation of the corresponding actions. To cope with both these issues, we propose a linear space-time warping technique with which we model variations in time using a time-scaling parameter a and a time-shifting parameter b. In addition, in order to achieve invariance against scaling, we introduce a scaling parameter c in the proposed warping technique. To accommodate this procedure, we propose the Chamfer distance as an appropriate distance measure, in order to cope with unequal number of features between different sets of salient points. More specifically, for two feature sets $F = \{(x_i, y_i, t_i), 1 \leq i \leq M\}$ and $F' = \{(x'_j, y'_j, t'_j), 1 \leq j \leq M'\}$ consisting of an M and M' number of features, respectively, the Chamfer distance of the set F from the set F' is defined as follows:

$$D(F, F') = \frac{1}{M} \sum_{i=1}^{M} \min_{j=1}^{M'} \sqrt{(x'_j - x_i)^2 + (y'_j - y_i)^2 + (t'_j - t_i)^2}\,. \tag{9}$$

From eq. 9 it is obvious that the selected distance measure is not symmetrical, as $D(F, F') \neq D(F', F)$. For recognition purposes, it is desirable to select a distance measure that is symmetrical. A measure that satisfies this requirement is the average of $D(F, F')$ and $D(F', F)$, that is,

$$D_c(F, F') = \frac{1}{2}(D(F, F') + D(F', F))\,. \tag{10}$$

Let us denote by $F_w = \{(cx_i, cy_i, at_i - b), 1 \leq i \leq M\}$ the feature set F with respect to feature set F'. Then, the distance between F' and F_w is given by eq. 9 as:

$$D(F_w, F') = \frac{1}{M} \sum_{i=1}^{M} \min_{j=1}^{M'} \sqrt{(x'_j - cx_i)^2 + (y'_j - cy_i)^2 + (t'_j - at_i + b)^2}\,. \tag{11}$$

Similarly, the feature set F' with respect to feature set F can be represented as $F'_w = \{(\frac{1}{c}x'_j, \frac{1}{c}y'_j, \frac{1}{a}t'_j + b), 1 \leq j \leq M'\}$ and their distance as:

$$D(F'_w, F) = \frac{1}{M'} \sum_{j=1}^{M'} \min_{i=1}^{M} \sqrt{(x_i - \frac{1}{c}x'_j)^2 + (y_i - \frac{1}{c}y'_j)^2 + (t_i - \frac{1}{a}t'_j - b)^2}\,. \tag{12}$$

The distance to be optimized follows from the substitution of eq. 11 and eq. 12 to eq. 10. We follow an iterative gradient descent approach for the adjustment of the a, b and c parameters. The update rules are given by:

$$a^{n+1} = a^n - \lambda_1 \frac{\partial D_c}{\partial a^n}, \tag{13}$$

$$b^{n+1} = b^n - \lambda_2 \frac{\partial D_c}{\partial b^n}, \tag{14}$$

$$c^{n+1} = c^n - \lambda_3 \frac{\partial D_c}{\partial c^n}\,, \tag{15}$$

where $\lambda_1, \lambda_2, \lambda_3$ are the learning rates and n is the iteration index. The algorithm iteratively adjusts the values of a, b and c towards the minimization of the Chamfer distance between the two feature sets, given by eq. 10. The iterative procedure stops when the values of a, b and c do not change significantly or after a fixed number of iterations.

4 Tracking

4.1 Auxiliary Particle Filtering

Recently, particle filtering tracking schemes [8], [16], have been successfully used [52], [53], [17] in order to track the state of a temporal event given a set of noisy observations. Its ability to maintain simultaneously multiple solutions, called particles, makes it particularly attractive when the noise in the observations is not Gaussian and makes it robust to missing or inaccurate data.

The particle filtering tracking scheme described in this section is initialized at the spatiotemporal salient points that are detected using the procedure of section 3. Let c denote the template that contains the color information in a rectangular window centered at each detected salient point and α denote the unknown location of the facial feature at the current time instant. Furthermore, let us denote by $Y = \{y^1, \ldots, y^-, y\}$ the observations up to the current time instant. The main idea of the particle filtering is to maintain a particle based representation of the a posteriori probability $p(\alpha|Y)$ of the state α given all the observations Y up to the current time instance. The distribution $p(\alpha|Y)$ is represented by a set of pairs (s_k, π_k) such that if s_k is chosen with probability equal to π_k, then it is as if s_k was drawn from $p(\alpha|Y)$. Our knowledge about the a posteriori probability is updated in a recursive way. Suppose that we have a particle based representation of the density $p(\alpha^-|Y^-)$, that is we have a collection of K particles and their corresponding weights (i.e. (s_k^-, π_k^-)). Then, the Auxiliary Particle Filtering can be summarized as follows:

1. Propagate all particles s_k^- via the transition probability $p(\alpha|\alpha^-)$ in order to arrive at a collection of K particles μ_k.
2. Evaluate the likelihood associated with each particle μ_k, that is let $\lambda_k = p(y|\mu_k; c)$.
   For the definition of $p(y|\mu_k; c)$ we use, in this paper, the observation model described in [17].
3. Draw K particles s_k^- from the probability density that is represented by the collection $(s_k^-, \lambda_k \pi_k^-)$. In this way, the auxiliary particle filter favors particles with high λ_k, that is particles which, when propagated with the transition density, end up at areas with high likelihood.
4. Propagate each particle s_k^- with the transition probability $p(\alpha|\alpha^-)$ in order to arrive at a collection of K particles ${s_k}'$.
5. Assign a weight ${\pi_k}'$ to each particle as follows,

$$ {w_k}' = \frac{p(y|{s_k}'; c)}{\lambda_k}, \quad {\pi_k}' = \frac{{w_k}'}{\sum_j w_j} \tag{16} $$

   This results in a collection of K particles and their corresponding weights (i.e. $\{({s_k}', {\pi_k}')\}$) which is an approximation of the density $p(\alpha|Y)$.

4.2 Online Background Estimation

The particle filtering tracking scheme described in the previous section is initialized at the spatiotemporal salient points that are detected using the procedure described in section 3. As indicated from eq. 1, the input signal that is used is the convolution of the original image sequence with a Gaussian derivative filter along the temporal dimension. The result of this is that the detected salient points are localized on the edges of the moving objects existing in the scene, rather than on the objects themselves. This fact may deteriorate the output of the tracker used, since the patches of the sequence that are being tracked also include a considerable portion of the scene's background. For this reason, we implement the adaptive background estimation algorithm described in [48], in order to determine which pixels belong to the foreground and which ones to the background. According to this algorithm, the values of a particular pixel over time are considered as a temporal process. At each time t, what is known about a particular pixel (x_0, y_0) is its history:

$$\{X_1, \ldots, X_t\} = \{I(x_0, y_0, i) : 1 \leq i \leq t\}, \tag{17}$$

where I is the image sequence. The recent history of each pixel is modeled by a mixture of K Gaussian distributions. The probability of observing the current pixel value is given by:

$$P(X_t) = \sum_{i=1}^{K} w_{i,t} \cdot \eta(X_t, \mu_{i,t}, \Sigma_{i,t}), \tag{18}$$

where K is the number of distributions, $w_{i,t}$ is an estimate of the weight of the i_{th} Gaussian in the mixture at time t, $\mu_{i,t}$ is the mean value of the i_{th} Gaussian in the mixture at time t, $\Sigma_{i,t}$ is the covariance matrix of the i_{th} Gaussian in the mixture at time t, and η is a Gaussian probability density function. K was set to 3, and the covariance matrix Σ is assumed to be diagonal, meaning that the RGB values of the pixels are assumed to be uncorrelated.

The parameters of each Gaussian mixture were initially estimated using the Expectation-Maximization (EM) algorithm and by using a small portion of the available data (i.e. the first few frames of the image sequence). Subsequently at each new frame t we follow an update procedure similar to the one of [48]. Every new pixel value X_t is checked against the existing K distributions until a match is found. A match is defined if the current pixel is within 3 standard deviations of a distribution. In case a match is found the parameters of the Gaussians are updated. If none of the K distributions match the current pixel value, the least probable distribution is replaced with a distribution with the current value as its mean value, an initially high variance, and low prior weight.

At each iteration of the particle filtering tracking scheme of section 4.1, every new particle is evaluated based on an invariant colour distance between the initial template (centered at the initializing spatiotemporal salient point) and the block that corresponds to the particle that is being evaluated. In order to take the estimated background model into account, we add an additional cost in

Fig. 2. Initial estimation of the background for an action where the subject is just raising its right hand

the evaluation process of each new particle. The additional cost for every pixel is equal to the probability that the pixel belongs to the current background model, that is,

$$C_{i,j,t} = \sum_{i=1}^{K} w_{i,j}\eta(X_{j,t}, \mu_{i,j,t}, \Sigma_{i,j,t}), \tag{19}$$

where K is the number of distributions, $w_{i,j,t}$ is an estimate of the weight of the i_{th} Gaussian in the mixture for the pixel j at time t, $\mu_{i,j,t}$ is the mean value of the i_{th} Gaussian in the mixture for the pixel j at time t and $\Sigma_{i,j,t}$ is the covariance matrix of the i_{th} Gaussian in the mixture for pixel j at time t.

If a pixel in the block belongs to the background, then eq. 19 will assign a large cost to that pixel, since the resulting probability will be high. If most pixels in the block belong to the background, then the additional cost to that block will also be large and consequently, a smaller weight will be assigned to it by the particle filter. In this way, the tracking scheme favors blocks that contain larger number of foreground pixels and assigns larger weights to the corresponding particles.

In Fig. 2 the initial background model that was estimated for an action where the subject is raising its right hand is presented. As can be seen from the figure, parts of the body that do not present significant motion are also considered part of the background. On the other hand, fast moving parts (e.g. right hand) are considered to belong to the foreground and are not included in the estimation.

5 Recognition

5.1 Longest Common Subsequence (LCSS) Algorithm

Using the analysis of the previous sections, we represent a given image sequence by a set of short trajectories, where each trajectory is initialized at a point which is considered salient both in space and time. Formally, an image sequence is represented by a set of trajectories $\{A_i\}, i = 1 \dots K$, where K is the number of trajectories that consist the set. Each trajectory is defined as $A_i = ((t_{i,n}, x_{i,n}, y_{i,n}), \dots)$, $n = 1 \dots N$, where $t_{i,n}, x_{i,n}, y_{i,n}$ are spatiotemporal coordinates and N is the number of samples that consist A_i. Let us define another trajectory set $\{B_j\}$, $j = 1 \dots L$ representing a different image sequence. Similar to $\{A_i\}$, the trajectories in $\{B_j\}$ are defined as $B_j = ((t_{j,m}, x_{j,m}, y_{j,m}), \dots)$, $m = 1 \dots M$, where M is the number of individual trajectories that consist $\{B_j\}$. We use a variant of the LCSS algorithm presented at [49], [50] in order to compare the two sets. Before we proceed with the comparison, we align the two sets in space and time using the a, b and c parameters that were computed using the procedure of section 3.3. Let us define the function $Head(A_i) = ((t_{i,n}, x_{i,n}, y_{i,n})), n = 1 \dots N-1$, that is, the individual trajectory A_i reduced by one sample. Then, according to the LCSS algorithm, the distance between individual trajectories A_i and B_j is given by:

$$d_L(A_i, B_j) = \begin{cases} 0, \quad if\ A_i\ or\ B_j\ is\ empty \\ \\ d_e((t_{i,n}, x_{i,n}, y_{i,n}), (t_{j,m}, x_{j,m}, y_{j,m})) \\ +d_L(Head(A_i), Head(B_j)), \\ if\ \ |t_{i,n} - t_{j,m}| < \delta\ and\ \ |x_{i,n} - x_{j,m}| < \varepsilon \\ and\ \ |y_{i,n} - y_{j,m}| < \varepsilon \\ \\ \max(d_L(Head(A_i), B_j), d_L(A_i, Head(B_j))) + p, \\ otherwise \end{cases}, \tag{20}$$

where d_e is the Euclidean distance, δ controls how far in time we can go in order to match a given point from one trajectory to a point in another trajectory, ϵ is the matching threshold and p is a penalty cost in case of mismatch. The notion of the LCSS distance of eq. 20 is depicted in Fig. 3.

Subsequently, the distance between sets $\{A_i\}$ and $\{B_j\}$is defined as follows:

$$D_L(\{A_i\}, \{B_j\}) = \frac{1}{K} \sum_i \min_j d_L(A_i, B_j) + \frac{1}{L} \sum_j \min_i d_L(B_j, A_i), \tag{21}$$

that is, the average over the set of the minimum distances, as they have been defined in eq. 20, between the K trajectories of set $\{A_i\}$ and the L trajectories of set $\{B_j\}$.

5.2 Relevance Vector Machine Classifier

We propose a classification scheme based on Relevance Vector Machines [51] in order to classify given examples of human actions. A Relevance Vector Machine (RVM) is a probabilistic sparse kernel model identical in functional form to the Support Vector Machines (SVM). In their simplest form, Relevance Vector

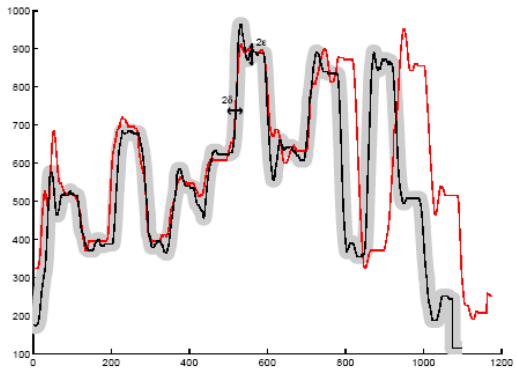


Fig. 3. The notion of the LCSS matching within a region of δ and ϵ of a trajectory.

Machines attempt to find a hyperplane defined as a weighted combination of a few Relevance Vectors that separate samples of two different classes. In contrast to SVM, predictions in RVM are probabilistic. Given a dataset of N input-target pairs $\{(F_n, l_n), 1 \leq n \leq N\}$, an RVM learns functional mappings of the form:

$$y(F) = \sum_{n=1}^{N} w_n K(F, F_n) + w_0 \,, \tag{22}$$

where $\{w_n\}$ are the model weights and $K(.,.)$ is a Kernel function. Gaussian or Radial Basis Functions have been extensively used as kernels in RVM. In our case, we use as a kernel a Gaussian Radial Basis Function defined by the distance measure of eq. 21. That is,

$$K(F, F_n) = e^{-\frac{D_L(F,F_n)^2}{2\eta}} \,, \tag{23}$$

where η is the Kernel width. RVM performs classification by predicting the posterior probability of class membership given the input F. The posterior is given by wrapping eq. 22 in a sigmoid function, that is:

$$p(l|F) = \frac{1}{1 + e^{-y(F)}} \tag{24}$$

In the two class problem, a sample F is classified to the class $l \in [0, 1]$, that maximizes the conditional probability $p(l|F)$. For L different classes, L different classifiers are trained and a given example F is classified to the class for which the conditional distribution $p_i(l|F)$,$1 \leq i \leq L$ is maximized, that is:

$$Class(F) = \arg\max_i (p_i(l|F)) \,. \tag{25}$$

6 Experimental Results

For the evaluation of the proposed method, we use aerobic exercises as a test domain. Our dataset consists of 12 different aerobic exercises, performed by amateurs, that have seen a video with an instructor performing the same set of exercises. Each exercise is performed twice by four different subjects, leading to a set of 96 corresponding feature sets.

In order to illustrate the ability of the proposed method to extract the kind of motion performed, we present in Fig. 4 the trajectories that were extracted from two different actions along with a snapshot of the corresponding actions. The salient points that are visible in the upper part of the figure were used in order to extract some of the trajectories presented in the lower part of the same Figure. Furthermore, the extracted trajectory set seems to correctly capture the pattern of the motion performed. This can easily be observed from the arch-like trajectories of the lower part of the figure, which correspond to the motion of the subjects' hands.

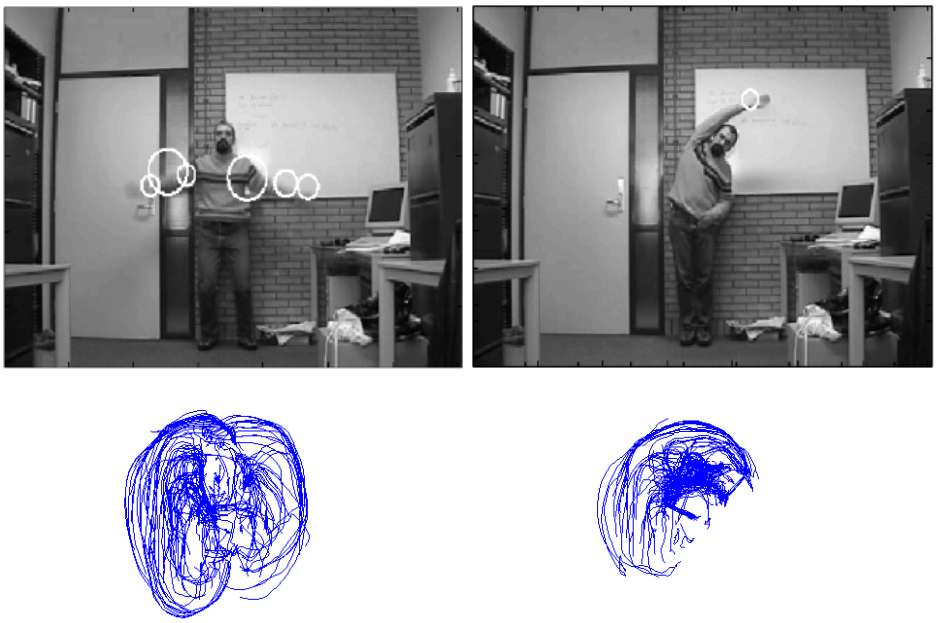


Fig. 4. Extracted trajectories for two different actions.

In order to classify a test example using the Relevance Vector Machines, we constructed 12 different classifiers, one for each class, and we calculated for each

Class Labels	**1**	**2**	**3**	**4**	**5**	**6**
RVM Recall	1	1	1	1	0.5	0.5
RVM Precision	1	1	1	1	0.44	0.4
Class Labels	**7**	**8**	**9**	**10**	**11**	**12**
RVM Recall	1	0.88	0.63	0.63	0.88	1
RVM Precision	1	1	0.63	0.83	0.88	1

Table 1. Recall and Precision rates for the kNN and RVM classifiers

test example F the conditional probability $p_i(l|F), 1 \leq i \leq 12$. Each example was assigned to the class for which the corresponding classifier provided the maximum conditional probability, as depicted in eq. 25. Note that for estimating each of the $p_i(l|F)$, an RVM is trained by leaving out the example F as well as all other instances of the same exercise that were performed by the subject from F. The corresponding recall and precision rates, calculated as an average of all test trials, are given in Table 1. The total recognition rate is equal to 80.61%, which is a relatively good performance, given the small number of examples with respect to the number of classes, and the fact that the subjects were not trained. In Table 2 the confusion matrix generated by the RVM classifier is also given.

Class labels	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**Total**
1	**8**	0	0	0	0	0	0	0	0	0	0	0	8
2	0	**8**	0	0	0	0	0	0	0	0	0	0	8
3	0	0	**8**	0	0	0	0	0	0	0	0	0	8
4	0	0	0	**8**	0	0	0	0	0	0	0	0	8
5	0	0	0	0	**4**	5	0	0	0	0	0	0	9
6	0	0	0	0	4	**3**	0	0	3	0	0	0	10
7	0	0	0	0	0	0	**8**	0	0	0	0	0	8
8	0	0	0	0	0	0	0	**7**	0	0	0	0	7
9	0	0	0	0	0	0	0	1	**5**	2	0	0	8
10	0	0	0	0	0	0	0	0	0	**5**	1	0	6
11	0	0	0	0	0	0	0	0	0	1	**7**	0	8
12	0	0	0	0	0	0	0	0	0	0	0	**8**	8
Total	8	8	8	8	8	8	8	8	8	8	8	8	8

Table 2. RVM Confusion Matrix

The confusion matrix in Table 2 conceals the fact that for some of the misclassified examples the probability assigned by the RVM classifier to the correct matching move might be very close to the probability assigned to the move actually selected by the classifier. We used the average ranking percentile in order to extract this kind of information and to measure the overall matching quality of our proposed algorithm. Let us denote with r^{F_n} the position of the correct match for the test example $F_n, n = 1 \ldots N_2$, in the ordered list of N_1 match values. Rank r^{F_n} ranges from $r = 1$ for a perfect match to $r = N_1$ for the worst possible match. Then, the average ranking percentile is calculated as follows:

$$\overline{r} = \left(\frac{1}{N_2} \sum_{n=1}^{N_2} \frac{N_1 - r^{F_n}}{N_1 - 1} \right) 100\%. \tag{26}$$

Since our dataset consists of 96 test image sequences divided in 12 separate classes, it follows that $N_1 = 12$ and $N_2 = 96$. Each of the 12 match values are provided for each example by the 12 trained RVM classifiers. The average ranking percentile for the RVM classifier is 94.5%. Its high value shows that for the majority of the missclassified examples, the correct matches are located in the first positions in the ordered list of match values.

7 Conclusions

In this work, previous work on spatiotemporal saliency was enhanced in order to extract a number of short trajectories from given image sequences. Each detected spatiotemporal point was used in order to initialize a tracker based on auxiliary particle filtering. A background estimation model was also implemented and incorporated into the particle evaluation process, in order to deal with inadequate localization of the initialization points and to improve, thus, the performance of the tracker. A variant of the LCSS algorithm was used in order to compare different sets of trajectories. The derived LCSS distance was used in order to define a kernel for the RVM classifier that was used for recognition. We have illustrated the efficiency of our representation in recognizing human actions using as a test domain aerobic exercises. Finally, we presented results on real image sequences that illustrate the consistency in the spatiotemporal localization and scale selection of the proposed method.

Acknowledgements

This work has been partially supported by the Dutch-French Van Gogh program (project VGP-62-604) and the work of A. Oikonomopoulos has been supported by the Greek State Scholarship Foundation (IKY). The data set was collected while I. Patras was with the ISIS group at the University of Amsterdam.

References

1. Pantic, M., Pentland, A., Nijholt, A., Huang, T.: Human computing and machine understanding of human behavior: A survey. International Conference on Multimodal Interfaces (2006)
2. Bar-Shalom, Y., Fortmann, T.: Tracking and Data Association. Academic Press (1988)
3. Julier, S., Uhlmann, J.: Unscented filtering and nonlinear estimation. Proceedings of the IEEE **92**(3) (2004) 401–422
4. Wu, Y., Hu, D., Wu, M., Hu, X.: Unscented kalman filtering for additive noise case: Augmented versus nonaugmented. IEEE Signal Processing Letters **12**(5) (2005) 357–360
5. LaViola, J.: A comparison of unscented and extended Kalman filtering for estimating quaternion motion. Proceedings of the American Control Conference **3** (2003) 2435 – 2440

6. Zhang, Y., Ji, Q.: Active and dynamic information fusion for facial expression understanding from image sequences. IEEE Trans. Pattern Analysis and Machine Intelligence **27**(5) (2005) 699 – 714
7. Gu, H., Ji, Q.: Information extraction from image sequences of real-world facial expressions. Machine Vision and Applications **16**(2) (2005) 105 – 115
8. Isard, M., Blake, A.: Condensation conditional density propagation for visual tracking. International Journal of Computer Vision **29**(1) (1998) 5 – 28
9. Isard, M., Blake, A.: Icondensation: Unifying low-level and high-level tracking in a stochastic framework. European Conference on Computer Vision **29**(1) (1998) 893 – 908
10. Lichtenauer, J., Hendriks, M.R.E.: Influence of the observation likelihood function on particle filtering performance in tracking applications. Automatic Face and Gesture Recognition (2004) 767– 772
11. Chang, C., Ansari, R., Khokhar, A.: Multiple object tracking with kernel particle filter. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition **1** (2005) 566– 573
12. Schmidt, J., Fritsch, J., Kwolek, B.: Kernel particle filter for real-time 3D body tracking in monocular color images. Automatic Face and Gesture Recognition (2006) 567– 572
13. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-Based Object Tracking. IEEE Trans. Pattern Analysis and Machine Intelligence **25**(5) (2003) 564–577
14. Yang, C., Duraiswami, R., Davis, L.: Fast multiple object tracking via a hierarchical particle filter. Proc. IEEE Int. Conf. Computer Vision **1** (2005) 212– 219
15. Shan, C., Wei, Y., Tan, T., Ojardias, F.: Real time hand tracking by combining particle filtering and mean shift. Automatic Face and Gesture Recognition **1** (2004) 669– 674
16. Pitt, M., Shephard, N.: Filtering via simulation: auxiliary particle filtering. J. American Statistical Association **94** (1999) 590 –
17. Patras, I., Pantic, M.: Tracking deformable motion. IEEE International Conference on Systems, Man and Cybernetics (2005) 1066 – 1071
18. Patras, I., Pantic, M.: Particle filtering with factorized likelihoods for tracking facial features. Automatic Face and Gesture Recognition (2004) 97 – 102
19. Matthews, I., Ishikawa, T., Baker, S.: The template update problem. In Proceedings of the British Machine Vision Conference. (2003)
20. Jepson, A., Fleet, D., El-Maraghi, T.: Robust Online Appearance Models for Visual Tracking. IEEE Trans. Pattern Analysis and Machine Intelligence **25**(10) (2003) 1296–1311
21. Avidan, S.: Support Vector Tracking. IEEE Trans. Pattern Analysis and Machine Intelligence **26**(8) (2004) 1064–1072
22. Gavrila, D., Davis, L.: 3-D Model-Based Tracking of Humans in Action: A Multi-view Approach. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition (1996) 73–80
23. MacCormick, J., Isard, M.: Partitioned Sampling, Articulated Objects and Interface-Quality Hand Tracking. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition (2000) 3–19
24. Stenger, B., Thayananthan, A., Torr, P., Cipolla, R.: Model-Based Hand Tracking Using a Hierarchical Bayesian Filter. IEEE Trans. Pattern Analysis and Machine Intelligence **28**(5) (2006) 1372–1384
25. Chang, W., Chen, C., Hung, Y.: Appearance-guided particle filtering for articulated hand tracking. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition **1** (2005) 235– 242

26. Sigal, L., Bhatia, S., Roth, S., Black, M., M. Isard, M.: Tracking loose-limbed people. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition **1** (2004) 421–428
27. Wu, Y., Hua, G., Yu, T.: Tracking articulated body by dynamic Markov network. Proc. IEEE Int. Conf. Computer Vision **2** (2003) 1094– 1101
28. Han, T., Ning, H., Huang, T.: Efficient Nonparametric Belief Propagation with Application to Articulated Body Tracking. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition **1** (2006) 214– 221
29. Deutscher, J., Blake, A., North, B., Bascle, B.: Tracking through Singularities and discontinuities by random sampling. Proc. IEEE Int. Conf. Computer Vision **2** (1999) 1144–1149
30. Black, M., Jepson, A.: Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. International Journal of Computer Vision **26**(1) (1998) 63–84
31. Kato, M., Y.W.Chen, G.Xu: Articulated Hand Tracking by PCA-ICA Approach. Automatic Face and Gesture (2006) 329– 334
32. Deutscher, J., Blake, A., Reid, I.: Articulated body motion capture by annealed particle filtering. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition **2** (2000) 126–133
33. Zhao, T., Nevatia, R.: Tracking Multiple Humans in Crowded Environment. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition **2** (2004) 406–413
34. Lanz, O.: Approximate Bayesian Multibody Tracking. IEEE Trans. Pattern Analysis and Machine Intelligence **28**(9) (2006) 1436–1449
35. Okuma, K., Taleghani, A., de Freitas, N., Little, J., Lowe, D.: A boosted particle filter: Multitarget detection and tracking. European Conference on Computer Vision **3021** (2004) 28–39
36. Nguyen, H., Ji, Q., Smeulders, A.: Spatio-Temporal Context for Robust Multitarget Tracking. IEEE Trans. Pattern Analysis and Machine Intelligence **29**(1) (2007) 52–64
37. Berclaz, J., F.'Fleuret, Fua, P.: Robust People Tracking with Global Trajectory Optimization. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition (2006) 744– 750
38. Avidan, S.: Ensemble Tracking. IEEE Trans. Pattern Analysis and Machine Intelligence **29**(2) (2007) 261–271
39. Nguyen, H., Smeulders, A.: Robust Tracking Using Foreground-Background Texture Discrimination. International Journal of Computer Vision **69**(3) (2006) 277–293
40. Figueroa, P., Leitey, N., Barros, R., Brenzikofer, R.: Tracking markers for human motion analysis. Proc. of IX European Signal Processing Conf., Rhodes, Greece (1998) 941 – 944
41. Moeslund, T., Nrgaard, L.: A brief overview of hand gestures used in wearable human computer interfaces. Technical Report CVMT 03-02, ISSN 1601-3646 (2003)
42. Haralick, R., Shapiro, L.: Computer and Robot Vision II. Addison-Wesley (1993) Reading, MA.
43. Gilles, S.: Robust Description and Matching of Images. PhD thesis, University of Oxford (1998)
44. Kadir, T., Brady, M.: Scale saliency: a novel approach to salient feature and scale selection. International Conference on Visual Information Engineering (2000) 25 – 28

45. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as space-time shapes. Proc. IEEE Int. Conf. Computer Vision **2** (2005) 1395 – 1402
46. Zelnik-Manor, L., Irani, M.: Event-based analysis of video. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition **2** (2001) 123 – 130
47. Oikonomopoulos, A., Patras, I., Pantic, M.: Spatiotemporal Salient Points for Visual Recognition of Human Actions. IEEE Trans. Systems, Man and Cybernetics Part B **36**(3) (2005) 710 – 719
48. Stauffer, C.: Adaptive background mixture models for real-time tracking. Proceedings, IEEE Conference on Computer Vision and Pattern Recognition (1999) 246 – 252
49. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. Proc. International Conference on Data Engineering (2002) 673 – 684
50. Buzan, D., Sclaroff, S., Kollios, G.: Extraction and clustering of motion trajectories in video. Proceedings, International Conference on Pattern Recognition **2** (2004) 521 – 524
51. Tipping, M.: The Relevance Vector Machine. Advances in Neural Information Processing Systems (1999) 652 – 658
52. Su, C., Zhuang, Y., Huang, L., Wu, F.: A two-step approach to multiple facial feature tracking: Temporal particle filter and spatial belief propagation. Proc. IEEE Intl Conf. on Automatic Face and Gesture Recognition (2004) 433 – 438
53. Pantic, M., Patras, I.: Dynamics of Facial Expressions-Recognition of Facial Actions and their Temporal Segments from Face Profile Image Sequences. IEEE Trans. Systems, Man and Cybernetics Part B **36**(2) (2006) 433 – 449