Switching Linear Inverse-Regression Model for Tracking Head Pose

Vincent Drouard, Sileye Ba, Radu Horaud

Switching Linear Inverse-Regression Model for Tracking Head Pose*

Vincent Drouard[1], Silèye Ba[1,2], Radu Horaud[1]
[1]INRIA Grenoble Rhône-Alpes, France [2]VideoStitch, France
vincent.drouard@inria.fr

Abstract

We propose to estimate the head-pose angles (pitch, yaw, and roll) by simultaneously predicting the pose parameters from observed high-dimensional feature vectors, and tracking these parameters over time. This is achieved by embedding a Gaussian mixture of linear inverse-regression model into a dynamic Bayesian model. The use of a switching Kalman filter (SKF) enables a principled way of carrying out this embedding. The SKF governs the temporal predictive distribution of the pose parameters (modeled as continuous latent variables) conditioned by the discrete variables associated with the mixture of linear inverse-regression formulation. We formally derive the equations of the proposed switching linear regression model, we propose an approximation that is both identifiable and computationally tractable, we design an EM procedure to estimate the SKF parameters in closed-form, and we carry out experiments and comparisons with other methods using recently released datasets.

1. Introduction

Recent advances in computer vision have demonstrated the relevance of representing images and image regions with feature vectors lying in high-dimensional feature spaces, *e.g*. SIFT [20], HOG [7], SURF [3], and any of their variants, or CNN-based features which may be used in conjunction with regression [15, 34] and tracking [21]. The rationale of representing image regions with high-dimensional feature vectors is that the latter supposedly embed hidden information, such as identity or pose. For example, in the case of face analysis, one can infer both face recognition and face orientation from such features. In the case of face orientation, or head pose, the task consists of extracting a low-dimensional parameterization, *i.e*. pitch, yaw and roll, from the high-dimensional feature space – a parameterization of the head-pose manifold. Not surprisingly, some of the best performing head-pose estimation methods rely either on dimensionality reduction followed by regression, [35, 32, 16, 19, 4, 12, 36], or on high-dimensional-to-low-dimensional regression, *e.g*. [28, 22, 8, 10].

Nevertheless, these feature-based approaches estimate the head-pose parameters from one image and are not designed to track the parameters over an image sequence. The observed feature vectors contain more than just head pose information, *e.g*. variabilities in illumination, appearance, shape, identity, background, clutter, etc. Moreover, errors in face localization are inherent, *i.e*. the bounding-box needed to extract the feature vector is not always correctly aligned with the face itself. All these time-varying phenomena may induce large oscillations and inconsistencies in the estimation of the parameter values, yielding non-smooth temporal trajectories.

In this paper we propose to simultaneously predict head-pose parameters from observed feature vectors and to track these parameters over time, based on embedding regression into a dynamic Bayesian model. Without loss of generality, we adopt a HOG-based description of faces, hence we need to predict a low-dimensional output (head-pose parameters) from a high-dimensional input (HOG vectors). To solve the latter we train the regression of [8] which is a generative Gaussian mixture of linear regression model. The proposed dynamic model is based on the switching Kalman filter (SKF) formulation. The proposed SKF governs the temporal predictive distribution of the pose parameters (which are continuous latent variables) conditioned by the state variables (the discrete latent variables associated with the mixture of linear regression formulation). The rationale of plugging regression into a dynamic Bayesian model is that the latter filters the prediction of the former while taking full advantage of the rich generative regression formulation.

We formally derive a switching dynamic Bayesian model, we devise an approximation of this model that is both identifiable and computationally tractable, we design an EM procedure that estimate the parameters, and we carry

*Funding from the European Union FP7 ERC Advanced Grant VHIA (#340113) is greatly acknowledged.

out experiments and comparisons with other methods. The principle of the tracker is summarized in Fig. 1 and an example is shown in Fig. 2.

The remainder of the paper is organized as follows. Section 2 discusses the related work. Section 3 summarizes the mixture of linear regression method used to predict pose parameters form the observed data. Section 4 describes in detail the proposed dynamic Bayesian model and Section 5 formally derives closed-form formulas for estimating the model parameters, *i.e*. model training. Experiments and comparisons with other methods are described in Section 6. Section 7 draws some conclusions.[1]

2. Related Work

Head-pose tracking has been an actively investigated topic; head-pose estimation and tracking methods were surveyed [29]. Many approaches rely on extracting facial landmarks, on tracking these landmarks over the image sequence and on estimating a rigid transformation between consecutive images, *e.g*. [13, 39], or between consecutive image pairs, *e.g*. [41]. Similarly, [24] builds a face graph based on the landmarks and tracks this graph over the image sequence. Another landmark-based approach [42] consists of using a 3D model of a generic face that embeds model-centered coordinates of facial landmarks, *e.g*. nose tip, eyes, lip corners, etc. The model is first fitted to the face detected in the first image and then fitted to the subsequent faces by tracking the landmarks. These methods heavily rely on landmark detection and tracking as well as on the robust estimation of the 2D-landmark-to-3D-landmark rigid transformation, *i.e*. the pose parameters. Therefore these methods are limited to frontal views of faces, because the landmarks are partially or totally occluded in side views of faces. Moreover, they track the facial landmarks instead of the pose parameters, hence they do not yield smooth pose trajectories. The advantage of the proposed method is that it relies neither on facial landmark detection nor on landmark tracking. The proposed method, once trained based on pairs of HOG descriptors and pose parameters, can deal with side views of faces, unlike landmark-based methods.

Head-pose tracking was also addressed using sampling methods based on particle filters, which allow to sample the temporal predictive distribution *e.g*. [2]. A principled way of combining a latent-variable temporal filter with the observed data is an important issue. In [38] it is proposed to extract a high-dimensional feature vector from a face and then to apply PCA to reduce its dimensionality. This assumes that the high-dimensional to low-dimensional mapping is linear (which may not be the case) and it does not guarantee that the PCA output contains pose information. Particle filtering can also be combined with a 3D deformable model and with facial landmarks, *e.g*. [9, 37]. As already outlined, landmark extraction is not always possible. The advantage of the proposed method over these particle-filter trackers is both theoretical and methodological: the feature-space to parameter-space mapping is combined with a dynamic model, and the estimation of the model parameters yields closed-form EM procedures. Moreover, the proposed SKF approximation, which makes the model computationally tractable, amounts to a variational approximation, thus yielding an extremely efficient runtime method.

Switching state space models have also been used to solve tracking problems. For example, [14], [30] and [18] show that the use of switching linear models helps tracking. In [31] switching models are applied for tracking people in videos in order to obtain motion-capture data, and three different approaches for inferring the parameters are compared, namely the Viterbi algorithm, variational inference, and the generalized pseudo Bayesian algorithm of order 2 (GPB2). The reported results obtained with these three approaches are quite similar. Viterbi has the lowest complexity, GPB2 yields the smoothest parameter trajectories, while the variational inference achieves a good compromise between low complexity and smooth trajectories.

The proposed method combines high-dimensional to low-dimensional mixture of linear regressions with a switching state-space model. In practice we adopt the GPB2 algorithm which, in combination with the generative regression model, yields closed-form expressions for the estimation of the tracked parameters. Hence, it is more efficient than sampling techniques which are often used in conjunction with generative tracking methods.

3. Mixture of Linear Inverse Regressions

In this section we summarize the mixture of linear *inverse* regressions of [8], which is named Gaussian locally linear mapping (GLLiM). GLLiM interchanges the roles of the input (high dimensional) and of the output (low dimensional), such that a *low-to-high* regression is being learned. The immediate consequence of this inverse regression strategy is a dramatic reduction in the number of model parameters, thus facilitating the task of training.

Let $\boldsymbol{X}$ and $\boldsymbol{Y}$ be two random variables, and let $\boldsymbol{x}$ and $\boldsymbol{y}$ denote their realizations, where $\boldsymbol{X} \in \mathbb{R}^L$ is the low-dimensional output (pose parameters) and $\boldsymbol{Y} \in \mathbb{R}^D$ ($D \gg L$) is the high-dimensional input (feature vectors). Once trained, the goal is to predict $\boldsymbol{x}$ given both an *input* $\boldsymbol{y}$ and

[1] Supplementary material for this paper can be found at `https://team.inria.fr/perception/research/head-pose-tracking/`

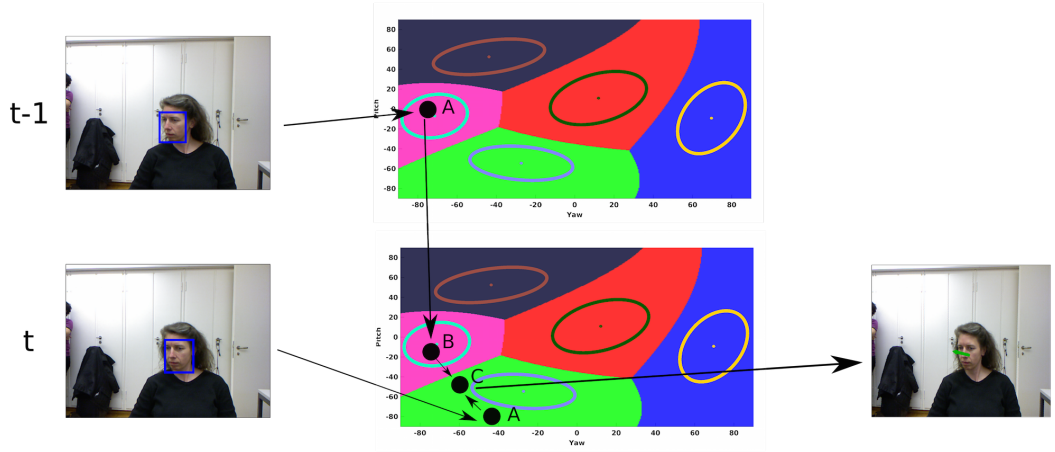

Figure 1. The method starts by learning a mixture of linear regression that allows the prediction of a head-pose from a HOG vector estimated from the bounding box of a face. Hence, Eq. (4) (Section 3) is applied at $t-1$ (top) and at t (bottom) and head poses are thus predicted, they are denoted A on the figure. Notice that, because of various perturbations in the data and of inherent flaws in face detection, the two predictions use two different affine transformations and hence they are associated with two different Gaussian components in the mixture, *e.g*. *magenta* and *green* on the figure. The proposed dynamic model combines the temporal prediction of the filter from $t-1$ to t, denoted B on the figure, with the pose predicted at t, to yield a filtered pose estimate, denoted C on the figure. The mixture of linear regression is plugged in the SKF model in a principled way.

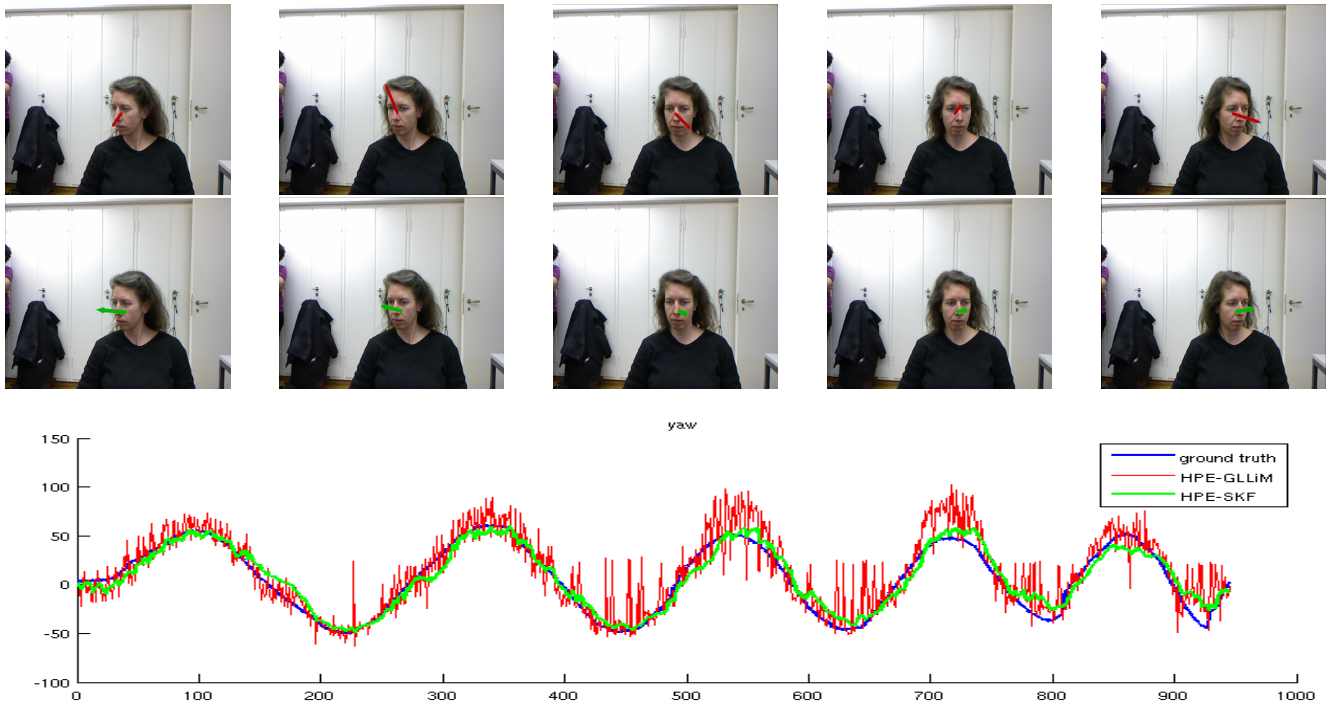

Figure 2. Yaw angles predicted with the mixture of linear regression method [10] (top sequence and red plot) and with the proposed method (bottom sequence and green plot). The ground-truth yaw trajectory is shown in blue.

the model parameters $\boldsymbol{\theta}$, *i.e*. $p(\boldsymbol{x}|\boldsymbol{y};\boldsymbol{\theta})$. We consider a *inverse low-to-high* regression, from the output variable $\boldsymbol{X}$ to the input variable $\boldsymbol{Y}$, *i.e*. a generative model, which is described by a mixture of locally-linear transformations, $\boldsymbol{y} = \sum_{k=1}^{K} \mathbb{I}(Z=k)(\mathbf{A}_k\boldsymbol{x} + \boldsymbol{b}_k + \boldsymbol{e}_k)$, where $\mathbb{I}$ is the indicator function, Z is a missing-data variable such that $Z=k$ if and only if $\boldsymbol{Y}$ is the image of $\boldsymbol{X}$ by the affine transformation $\boldsymbol{y} = \mathbf{A}_k\boldsymbol{x} + \boldsymbol{b}_k$, with $\mathbf{A}_k \in \mathbb{R}^{D\times L}$ and $\boldsymbol{b}_k \in \mathbb{R}^D$, and where $\boldsymbol{e}_k \in \mathbb{R}^D$ is an error vector capturing both the high-dimensional observation noise and the reconstruction error

due to the piecewise approximation of a non-linear function. The missing-data variable Z allows one to write the joint probability of $\boldsymbol{X}$ and $\boldsymbol{Y}$ as the following mixture:

$$p(\boldsymbol{y}, \boldsymbol{x}; \boldsymbol{\theta}) = \sum_{k=1}^{K} p(\boldsymbol{y}|\boldsymbol{x}, Z = k; \boldsymbol{\theta}) \times p(\boldsymbol{x}|Z = k; \boldsymbol{\theta}) p(Z = k; \boldsymbol{\theta}), \quad (1)$$

where $\boldsymbol{\theta}$ denotes the model parameters. Assuming that $\boldsymbol{e}_k$ is a zero-mean Gaussian variable with a diagonal covariance matrix $\boldsymbol{\Sigma}_k \in \mathbb{R}^{D \times D}$, we obtain that

$$p(\boldsymbol{y}|\boldsymbol{x}, Z = k; \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{y}; \mathbf{A}_k \boldsymbol{x} + \boldsymbol{b}_k, \boldsymbol{\Sigma}_k). \quad (2)$$

If we further assume that $\boldsymbol{X}$ follows a mixture of Gaussians via the same assignment $Z = k$, we can write that

$$p(\boldsymbol{x}|Z = k; \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{c}_k, \boldsymbol{\Gamma}_k),\ p(Z = k; \boldsymbol{\theta}) = \pi_k, \quad (3)$$

where $\boldsymbol{c}_k \in \mathbb{R}^L$, $\boldsymbol{\Gamma}_k \in \mathbb{R}^{L \times L}$ and $\sum_{k=1}^{K} \pi_k = 1$. Note that this representation induces a partition of $\mathbb{R}^L$ into K regions $\mathcal{R}_k$, where $\mathcal{R}_k$ is the region where the transformation $(\mathbf{A}_k, \boldsymbol{b}_k)$ is most likely invoked, *e.g.* Fig. 1. This model is described by the *inverse* parameter set $\boldsymbol{\theta} = \{\boldsymbol{c}_k, \boldsymbol{\Gamma}_k, \pi_k, \mathbf{A}_k, \boldsymbol{b}_k, \boldsymbol{\Sigma}_k\}_{k=1}^{K}$. The model parameters can be estimated via an EM algorithm. The expectation step computes the responsibilities, namely $p(Z_n = k|\boldsymbol{x}_n, \boldsymbol{y}_n; \boldsymbol{\theta}^{(\text{old})})$, given the old parameter values $\boldsymbol{\theta}^{(\text{old})}$, while the maximization step computes new parameter values via maximization of the expected complete-data log-likelihood function, namely $\boldsymbol{\theta}^{(\text{new})} = \text{argmax}\, \mathbb{E}[\log p(\boldsymbol{x}, \boldsymbol{y}, Z|\boldsymbol{\theta}^{(\text{old})})]$, which yields a closed-form solution [8]. Initial responsibilities are obtained by fitting a K-component GMM to the low-dimensional data $\{\boldsymbol{x}_n\}_{n=1}^{N}$.

Once the inverse parameter vector $\boldsymbol{\theta}$ is estimated one obtains a *low-dimensional to high-dimensional* inverse predictive distribution [8]. The *high-dimensional to low-dimensional* forward predictive distribution has a closed-form expression:

$$p(\boldsymbol{x}|\boldsymbol{y}; \boldsymbol{\theta}^*) = \sum_{k=1}^{K} \frac{\pi_k^* \mathcal{N}(\boldsymbol{y}; \boldsymbol{c}_k^*, \boldsymbol{\Gamma}_k^*)}{\sum_{j=1}^{K} \pi_j^* \mathcal{N}(\boldsymbol{y}; \boldsymbol{c}_j^*.\boldsymbol{\Gamma}_j^*)} \mathcal{N}(\boldsymbol{x}; \mathbf{A}_k^* \boldsymbol{y} + \boldsymbol{b}_k^*, \boldsymbol{\Sigma}_k^*), \quad (4)$$

which is a Gaussian mixture fully defined by the forward parameters $\boldsymbol{\theta}^* = \{\boldsymbol{c}_k^*, \boldsymbol{\Gamma}_k^*, \pi_k^*, \mathbf{A}_k^*, \boldsymbol{b}_k^*, \boldsymbol{\Sigma}_k^*\}_{k=1}^{K}$, that can be obtained in analytically from the inverse parameters:

$$\boldsymbol{c}_k^* = \mathbf{A}_k \boldsymbol{c}_k + \boldsymbol{b}_k,\ \boldsymbol{\Gamma}_k^* = \boldsymbol{\Sigma}_k + \mathbf{A}_k \boldsymbol{\Gamma}_k \mathbf{A}_k^\top,\ \pi_k^* = \pi_k, \quad (5)$$

$$\mathbf{A}_k^* = \boldsymbol{\Sigma}_k^* \mathbf{A}_k^\top \boldsymbol{\Sigma}_k^{-1},\ \boldsymbol{b}_k^* = \boldsymbol{\Sigma}_k^* (\boldsymbol{\Gamma}_k^{-1} \boldsymbol{c}_k - \mathbf{A}_k^\top \boldsymbol{\Sigma}_k^{-1} \boldsymbol{b}_k), \quad (6)$$

$$\boldsymbol{\Sigma}_k^* = (\boldsymbol{\Gamma}_k^{-1} + \mathbf{A}_k^\top \boldsymbol{\Sigma}_k^{-1} \mathbf{A}_k)^{-1}. \quad (7)$$

4. The Dynamic Bayesian Model

The main difference between the probabilistic regression model outlined in Section 3 and the proposed temporal model is that the conditional distribution $p(\boldsymbol{x}|\boldsymbol{y})$ is replaced with $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$, where t is the time index. The proposed graphical model is shown on Fig. 3, where Z_t is the discrete latent variable associated with the Gaussian mixture of linear regression, $\boldsymbol{X}_t$ and $\boldsymbol{Y}_t$ are the latent head pose and the observed high-dimensional feature vector at t, respectively. Using Bayes rule and marginalization we obtain:

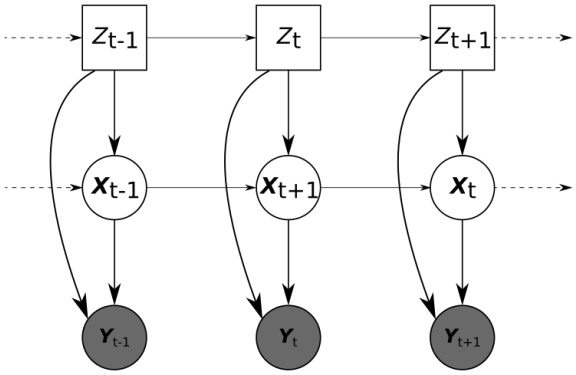

Figure 3. Proposed graphical model

$$p(\boldsymbol{X}_t = \boldsymbol{x}_t|\boldsymbol{Y}_{1:t} = \boldsymbol{y}_{1:t}) = \sum_{j=1}^{K} \sum_{i=1}^{K} \int_{\boldsymbol{X}_{t-1}} \frac{1}{p(\boldsymbol{y}_t)} \times p(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, Z_t = j, Z_{t-1} = i, \boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}) d\boldsymbol{X}_{t-1}. \quad (8)$$

Under the Markovian assumption and using the conditional independencies associated with the proposed graphical model of Fig. 3, we can write the term inside the integral of (8) as follows:

$$\begin{aligned} &p(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, Z_t = j, Z_{t-1} = i, \boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}) \\ &\quad = p(\boldsymbol{y}_t|\boldsymbol{x}_t, Z_t = j)\, p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, Z_t = j) \\ &\quad \times p(Z_t = j|Z_{t-1} = i)\, p\left(\boldsymbol{x}_{t-1}, Z_{t-1} = i|\boldsymbol{y}_{1:t-1}\right). \end{aligned} \quad (9)$$

The first probability on the right hand side of this equation, $p(\boldsymbol{y}_t|\boldsymbol{x}_t, Z_t = j)$ is the Gaussian distribution introduced in (2). The main difference between the *static* model and the *dynamic* model is that (3) is replaced with:

$$p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, Z_t = j) = \mathcal{N}(\boldsymbol{x}_t|\mathbf{C}_j \boldsymbol{x}_{t-1}, \mathbf{Q}_j), \quad (10)$$

$$p(Z_t = j|Z_{t-1} = i) = \tau_{ij}. \quad (11)$$

The parameters of the temporal model will be jointly denoted by $\boldsymbol{\phi}$:

$$\boldsymbol{\phi} = \{\mathbf{C}_j, \mathbf{Q}_j, \tau_{ij}, i, j = 1 \dots K\}. \tag{12}$$

By substituting (2), (10), and (11) into (9), by using basic properties of Gaussian distributions (Gaussian product and Gaussian integral), e.g. [6], and after some derivations, one can show that (8) can be written as a K^2-component GMM:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}; \boldsymbol{\psi}_{t|t-1}) = \sum_{i=1}^{K}\sum_{j=1}^{K} \pi_{t|t-1}^{ij} \mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\mu}_{t|t-1}^{ij}, \mathbf{W}_{t|t-1}^{ij}), \tag{13}$$

which in turn can be approximated with another K-component GMM, namely:

$$p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}; \boldsymbol{\lambda}_t) \approx \sum_{j=1}^{K} \rho_t^j \mathcal{N}(\boldsymbol{x}_t|\boldsymbol{\eta}_t^j, \mathbf{V}_t^j). \tag{14}$$

The parameters of the these two Gaussian mixtures are denoted with

$$\boldsymbol{\psi}_{t|t-1} = \{\pi_{t|t-1}^{ij}, \boldsymbol{\mu}_{t|t-1}^{ij}, \mathbf{W}_{t|t-1}^{ij}, i, j = 1 \dots K\} \tag{15}$$

and with

$$\boldsymbol{\lambda}_t = \{\rho_t^j, \boldsymbol{\eta}_t^j, \mathbf{V}_t^j, j = 1 \dots K\}. \tag{16}$$

The K-component GMM approximation (13) of the K^2-component GMM (14) is necessary in order to avoid an exponential grow of the number of components, hence it guarantees the computational tractability of the temporal model. As discussed in [27], three approaches have been proposed to avoid the number of components to explode. Our approximation is based on the generalized pseudo Bayesian algorithm (GPBa) of order 2 (GPB2) which, according to [31], yields a smooth output.

One interesting consequence of replacing (3) with (10) is that the parameter set $\boldsymbol{\theta}$ in Section 3 is replaced with a reduced parameter set $\boldsymbol{\theta}_r = \{\mathbf{A}_j, \boldsymbol{b}_j, \boldsymbol{\Sigma}_j\}_{j=1}^K$. Consequently, the formulae in (5) and (6) are simplified:

$$\mathbf{A}_j^* = \boldsymbol{\Sigma}_j^* \mathbf{A}_j^\top \boldsymbol{\Sigma}_j^{-1}, \quad \boldsymbol{b}_j^* = -\mathbf{A}_j^* \boldsymbol{b}_j, \quad \boldsymbol{\Sigma}_j^* = (\mathbf{A}_j^\top \boldsymbol{\Sigma}_j^{-1} \mathbf{A}_j)^{-1}. \tag{17}$$

It can be shown that the parameter set (15) can be written as a function of $\boldsymbol{\theta}_r, \boldsymbol{\lambda}_{t-1}$ and $\boldsymbol{\phi}$:

$$\mathbf{W}_{t|t-1}^{ij} = \left(\boldsymbol{\Sigma}_j^{*-1} + \mathbf{P}_{t-1}^{ij}\right)^{-1}, \tag{18}$$

$$\boldsymbol{\mu}_{t|t-1}^{ij} = \mathbf{W}_{t|t-1}^{ij} \left(\boldsymbol{\Sigma}_j^{*-1} \left(\mathbf{A}_j^* \boldsymbol{y}_t + \boldsymbol{b}_j^*\right) + \mathbf{P}_{t-1}^{ij} \mathbf{C}_j \boldsymbol{\eta}_{t-1}^j\right), \tag{19}$$

$$\pi_{t|t-1}^{ij} \propto r_{t|t-1}^{ij} = \rho_{t-1}^i \tau_{ij} \mathcal{N}(\boldsymbol{d}_{t|t-1}^{ij}|0, \mathbf{S}_{t|t-1}^{ij}), \tag{20}$$

where $\mathbf{P}_{t-1}^{ij}$, $\boldsymbol{d}_{t|t-1}^{ij}$, and $\mathbf{S}_{t|t-1}^{ij}$ are defined by:

$$\mathbf{P}_{t-1}^{ij} = \left(\mathbf{Q}_j + \mathbf{C}_j \mathbf{V}_{t-1}^i \mathbf{C}_j^\top\right)^{-1}, \tag{21}$$

$$\boldsymbol{d}_{t|t-1}^{ij} = \mathbf{y}_t - \mathbf{A}_j(\mathbf{C}_j \boldsymbol{\eta}_{t-1}^i) - \boldsymbol{b}_j, \tag{22}$$

$$\mathbf{S}_{t|t-1}^{ij} = \boldsymbol{\Sigma}_j + \mathbf{A}_j(\mathbf{Q}_j + \mathbf{C}_j \mathbf{V}_{t-1}^i \mathbf{C}_j^\top)\mathbf{A}_j^\top. \tag{23}$$

Eq. (21) defines the covariance of the prediction variable dynamics, Eq. (22) is the difference between the observation and the predicted observation, given $\boldsymbol{\eta}_{t-1}$, and Eq. (23) defines the associated covariance matrix.

The mean (19) can be seen as a "weighted" linear combination of the dynamical prediction $\mathbf{C}_j \eta_{t-1}^j$ and of the prediction based on observation $\mathbf{A}_j^* \boldsymbol{y}_t + \boldsymbol{b}_j^*$, where the "weights" are covariance matrices. Thus the confidence related to the covariance matrices defines the weights of the dynamical prediction and the observation prediction in the final estimation. Eq. (18) is the associated covariance matrix, which is the inverse of the sum of the precision matrix of the temporal prediction $\mathbf{P}_{t-1}^{ij}$ and precision matrix $\boldsymbol{\Sigma}_j^{*-1}$ of the observation $\boldsymbol{y}_t$. The GMM proportions in Eq. (20) are defined as a product between three terms: the proportions of the i^{th} components at $t-1$, ρ_{t-1}^i, the switching filter transition probabilities τ_{ij}, and $\mathcal{N}(\boldsymbol{d}_{t|t-1}^{ij}|0, \mathbf{S}_{t|t-1}^{ij})$.

Using the mixture reduction scheme explained in [33], the parameters of the K-component GMM $\boldsymbol{\lambda}_t$ can now be evaluated from the parameters of the K^2-component GMM $\boldsymbol{\psi}_{t|t-1}$, with the following formulas:

$$\boldsymbol{\eta}_t^j = \sum_{i=1}^{K} \tilde{\pi}_{t|t-1}^{ij} \boldsymbol{\mu}_{t|t-1}^{ij}, \tag{24}$$

$$\mathbf{V}_t^j = \sum_{i=1}^{K} \tilde{\pi}_{t|t-1}^{ij} \left(\mathbf{W}_{t|t-1}^{ij} + (\boldsymbol{\mu}_{t|t-1}^{ij} - \boldsymbol{\eta}_t^j)(\boldsymbol{\mu}_{t|t-1}^{ij} - \boldsymbol{\eta}_t^j)^\top\right), \tag{25}$$

$$\boldsymbol{\rho}_t^j = \sum_{i=1}^{K} \pi_{t|t-1}^{ij}, \text{ with } \tilde{\pi}_{t|t-1}^{ij} = \pi_{t|t-1}^{ij} / \sum_{k=1}^{k} \pi_{t|t-1}^{kj}. \tag{26}$$

5. Estimating the Model Parameters

This section describes the estimation of the parameters ϕ in (12) via learning (please consult [27] for a detailed description). We remind that the estimation of the regression parameters θ is described in detail in [8] and summarized in Section 3. We use an EM procedure to estimate the parameters $\left\{\mathbf{C}_j, \mathbf{Q}_j\right\}_{j=1}^K$. During the E-step we compute the complete-data expected log-likelihood. First, we express

the complete-data log-likelihood $\mathcal{L}$ as follows:

$$\begin{aligned}\mathcal{L} &= \log p\left(\boldsymbol{x}_{1:T}, Z_{1:T}, \boldsymbol{y}_{1:T}; \theta\right) \\ &= \sum_{t=1}^{T}\sum_{j=1}^{K} \delta(t,j) \log \mathcal{N}(\boldsymbol{y}_t; \mathbf{A}_j \boldsymbol{x}_t + \boldsymbol{b}_j, \boldsymbol{\Sigma}_j) \\ &+ \sum_{t=2}^{T}\sum_{j=1}^{K} \delta(t,j) \log \mathcal{N}(\boldsymbol{x}_t; \mathbf{C}_j \boldsymbol{x}_{t-1}, \mathbf{Q}_j) \\ &+ \sum_{t=2}^{T}\sum_{j=1}^{K}\sum_{i=1}^{K} \delta(t,j)\delta(t-1,i) \log \tau_{ij} + \log p\left(\boldsymbol{x}_1, Z_1\right),\end{aligned} \tag{27}$$

where $\delta(t,q)$ is equal to 1 if $Z_t = q$ and 0 otherwise. To complete the E-step we evaluate the above expected log-likelihood. The M-step maximizes it with respect to the parameters that must be estimated. Thus we obtain the following formulas for the estimation of the parameters ϕ:

$$\begin{aligned}\mathbf{C}_j = &\left(\sum_{t=2}^{T} p\left(Z_t = j|\boldsymbol{y}_t\right) \mathbb{E}\left[\boldsymbol{x}_t \boldsymbol{x}_{t-1}^T\right]\right) \\ &\times \left(\sum_{t=2}^{T} p\left(Z_t = j|\boldsymbol{y}_t\right) \mathbb{E}\left[\boldsymbol{x}_{t-1} \boldsymbol{x}_{t-1}^T\right]\right)^{-1},\end{aligned} \tag{28}$$

$$\begin{aligned}\mathbf{Q}_j = &\frac{1}{\sum_{t=2}^{T} p\left(Z_t = j|\boldsymbol{y}_t\right)} \\ &\times \left(\sum_{t=2}^{T} p\left(Z_t|\boldsymbol{y}_t\right) \left(\mathbb{E}\left[\boldsymbol{x}_t \boldsymbol{x}_t^T\right] - \mathbf{C}_j \mathbb{E}\left[\boldsymbol{x}_{t-1} \boldsymbol{x}_t^T\right]\right)\right),\end{aligned} \tag{29}$$

where:

$$\begin{aligned}\mathbb{E}\left[\boldsymbol{x}_t \boldsymbol{x}_{t-1}^T\right] &= \mathbf{V}_{t,t-1} + \boldsymbol{\eta}_t \boldsymbol{\eta}_{t-1}^\top, \\ \mathbb{E}\left[\boldsymbol{x}_{t-1} \boldsymbol{x}_{t-1}^T\right] &= \mathbf{V}_{t-1} + \boldsymbol{\eta}_{t-1} \boldsymbol{\eta}_{t-1}^\top.\end{aligned}$$

Parameters $\boldsymbol{\eta}_t$, $\boldsymbol{\eta}_{t-1}$, $\mathbf{V}_{t-1}$ and $\mathbf{V}_{t,t-1}$ are obtained using the smoothing statistics $p\left(\boldsymbol{x}_t|\boldsymbol{y}_{1:T}\right)$ which is expressed as follows:

$$\begin{aligned}p(\boldsymbol{x}_t|\boldsymbol{y}_{1:T}) &= \sum_{j=1}^{K}\sum_{i=1}^{K} p\left(\boldsymbol{x}_t, Z_t = j, Z_{t+1} = i|\boldsymbol{y}_{1:T}\right) \\ &= \sum_{j=1}^{K}\sum_{i=1}^{K} p\left(\boldsymbol{x}_t|Z_t = j, Z_{t+1} = i, \boldsymbol{y}_{1:T}\right) \\ &\quad \times p\left(Z_t = j, Z_{t+1} = i|\boldsymbol{y}_{1:T}\right),\end{aligned} \tag{30}$$

where

$$\begin{aligned}p(\boldsymbol{x}_t|Z_t = j, Z_{t+1} = i, \boldsymbol{y}_{1:T}) \\ = p\left(\boldsymbol{x}_t|Z_t = j, \boldsymbol{y}_{1:t}\right) p\left(\boldsymbol{y}_{t+1:T}|Z_{t+1} = i, \boldsymbol{x}_t\right).\end{aligned} \tag{31}$$

The first term of (31) $p\left(\boldsymbol{x}_t|Z_t = j, \boldsymbol{y}_{1:t}\right)$ is the forward recursion and is defined in Section 4 as $\mathcal{N}\left(\boldsymbol{x}_t|\eta_t^j, \mathbf{V}_t^j\right)$. The second term $p\left(\boldsymbol{y}_{t+1:T}|Z_{t+1} = i, \boldsymbol{x}_t\right)$ defines the backward recursion:

$$\begin{aligned}&p(\boldsymbol{y}_{t+1:T}|Z_{t+1} = i, \boldsymbol{x}_t) = \\ &\int_{\boldsymbol{X}_{t+1}} p\left(\boldsymbol{x}_{t+1}|\boldsymbol{x}_t, Z_{t+1} = i\right) \\ &\times p\left(\boldsymbol{y}_{t+1}|\boldsymbol{x}_{t+1}, Z_{t+1} = i\right) p\left(\boldsymbol{y}_{t+2:T}|\boldsymbol{x}_{t+1}\right) d\boldsymbol{X}_{t+1},\end{aligned} \tag{32}$$

where $p\left(\boldsymbol{y}_{t+2:T}|\boldsymbol{x}_{t+1}\right) = \mathcal{N}\left(\boldsymbol{x}_{t+1}|\boldsymbol{\eta}_{t+1}^b, \mathbf{V}_{t+1}^b\right)$. The second term of Eq. (30) can be decomposed as follows, using [17]:

$$\begin{aligned}&p\left(Z_t = j, Z_{t+1} = i|\boldsymbol{y}_{1:T}\right) \simeq \\ &p\left(Z_{t+1} = i|\boldsymbol{y}_{1:T}\right) \frac{p\left(Z_{t+1} = i|Z_t = j\right) p\left(Z_t = j|\boldsymbol{y}_{1:t}\right)}{p\left(Z_{t+1} = i|\boldsymbol{y}_{1:t}\right)},\end{aligned} \tag{33}$$

where:

$$p\left(Z_t = j|\boldsymbol{y}_{1:t}\right) = \rho_t^j \tag{34}$$

$$p\left(Z_{t+1} = i|\boldsymbol{y}_{1:t}\right) = \sum_{j=1}^{K} p\left(Z_{t+1} = i|Z_t = j\right) p\left(Z_t = j|\boldsymbol{y}_{1:t}\right) \tag{35}$$

$$p\left(Z_{t+1} = i|\boldsymbol{y}_{1:T}\right) = \sum_{j=1}^{K} p\left(Z_{t+1} = i, Z_{t+2} = j|\boldsymbol{y}_{1:T}\right) \tag{36}$$

As outlined in Section 4, the number of component increases (30), hence we merge the Gaussians twice: first over Z_{t+1} to obtain a mixture of K Gaussian components with mean $\boldsymbol{\eta}_t^{jb}$, covariance $\mathbf{V}_t^{jb}$ and proportions $p\left(Z_t = j|\boldsymbol{y}_{1:T}\right) = \rho_t^{jb}$, second over Z_t and thus we obtain a single Gaussian component, with mean $\boldsymbol{\eta}_t$ and covariance $\mathbf{V}_t$.

To estimate the transition matrix $\{\tau_{ij}\}_{i,j=1}^{K}$ we employ the Lagrange multiplier method to maximize the log-likelihood with respect to τ_{ij}, hence we obtain the following expression for the transition probabilities:

$$\tau_{ij} = \frac{\sum_{t=2}^{T} p\left(Z_t = j|\boldsymbol{y}_{1:T}\right) p\left(Z_{t-1} = i|\boldsymbol{y}_{1:T}\right)}{\sum_{t=2}^{T} p\left(Z_{t-1} = i|\boldsymbol{y}_{1:T}\right)} \tag{37}$$

6. Experiments

To gauge the performance of the proposed method we used two datasets: the Biwi-Kinect head-pose dataset [11] and the EYEDIAP dataset [25]. Biwi-Kinect comprises 24 videos of 20 different people (16 men and 4 women)

recorded with a Kinect camera. During the recordings people were asked to move their heads freely in front of the camera. 3D head pose (pitch, yaw, and roll angles) annotations are automatically and accurately provided for each video frame using the face-shift software. The angle values range from -60° to 60° for pitch, -75° to 75° for yaw and -20° to 20° for roll. The dataset provides RGB and depth images as well as the calibration matrices. The 3D nose positions are provided as well. EYEDIAP is a dataset for gaze and head-pose estimation. It provides 94 videos of 16 people recorded using different configurations, such as static and turning heads. The dataset provides RGB videos (in both HD and GVA quality) and depth videos with the associated calibrations matrices. For each video, annotations of both head-pose and gaze are provided for each frame. The angle values range from -40° to 40° for pitch and -50° to 50° for yaw. In our experiments we only used the RGB images where people are looking at a moving object (named A_FT_M in the dataset).

The proposed model, referred to as HPE_SKF (head-pose estimation based on SKF) is compared to the following methods: (i) a landmark-based approach that uses the facial landmark localization method of [39] (Flandmarks) combined with 2D-to-3D landmark-based pose estimation method, namely the PnP (perspective n-point) algorithm available with OpenCV, (ii) a depth head model based on method [26] that learns a 3D head model using 16 manually annotated facial landmarks on multiple frames to learn the model, and ICP (iterative closest point algorithm) to estimate the transformation (rotation and translation) of the head pose between two consecutive frames in order to track the pose over time, (iii) the regression-based method of [10] which is referred to as HPE-GLLiM, and (iv) the regression method [10] combined with a standard Kalman filter [1, 6]. Both (i) and (ii) perform tracking. For evaluation, on the Biwi-Kinect dataset we used the leave-one-out protocol: all the data related to one person are put aside and the remaining data of the other persons are used to train the model.

Using the EYEDIAP dataset, we compared our method against the baseline method of [26] which requires to learn a person based on his/her 3D head model. In this case we did not apply the leave-one-out protocol but instead we learned a person based head-pose regression model (θ) on a subsample of the frames associated with each person. Then, for each person we estimate the tracking parameters (ϕ) over the whole video. Thus, we obtain a person-based tracking model for head pose. As a measure of performance, we use the mean absolute deviation between the ground-truth and the estimated pose.

Face regions are extracted from images with a face detector [40]. This detector is efficient and robust with both frontal- and side-views of faces. Using the detection we run a face tracker using particle filtering to extract a face at each frames of the videos. Nevertheless, the obtained face regions are noisy, *i.e*. the bounding boxes are not always nicely aligned onto the faces. This yields extremely realistic input data for the tested methods, unlike other head-pose benchmarks that use manually extracted bounding boxes, *e.g*. using the nose tip as the bounding box center. From each face region thus detected, we extract HOG features with different cell resolutions (in a pyramid-like fashion), namely 32×32, 16×16 and 8×8 pixels, with block size of 2×2 cells and 8 bins to quantize the gradient orientation. This results in feature vectors of size $D = 1888$.

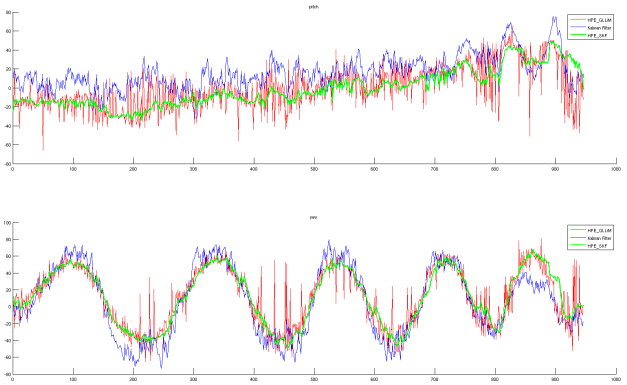

Figure 4. Comparison between the estimated pitch angle (top) and yaw angle (bottom) with three different methods: HPE_GLLiM (red), Kalman Filter (blue), and the proposed HPE_SKF (green) for the Biwi-Kinect dataset.

The regression parameters θ (Section 3) and the SKF parameters ϕ Eq. (12) are learned separately. First, the regression parameters θ are estimated using the EM algorithm described in [8]. Second, the filtering parameters ϕ are estimated using the method described in Section 5. The covariances are initialized with identity matrices and the transition matrix $\{\tau_{ij}\}_{i,j=1}^{K}$ is initialized with the Battacharyya distance [5] between two subspaces obtained using the parameters θ of the low dimensional space defined by Z.

Table 1. Average (Avg.) and standard deviation (Std.) of the absolute error (in degrees) for the pitch, yaw and roll angles (when applicable) on the Biwi-Kinect dataset. Head bounding boxes are extracted using a face detection algorithm.

	Pitch		Yaw		Roll	
Methods	Avg.	Std.	Avg.	Std.	Avg.	Std.
[39]	13.12	10.79	21.1	14.16	–	–
[10]	14.35	13.73	12.52	13.52	10.89	9.82
[1, 6]	27.43	17.61	15.13	12.09	14.62	9.94
HPE_SKF	**10.03**	**8.73**	**8.6**	**7.21**	**8.48**	**8.01**

Table 2. Average (Avg.) and standard deviation (Std.) of the absolute error (in degrees) for the pitch and yaw angles on the EYEDIAP dataset. Head bounding boxes are extracted using a face detection algorithm combined with a face tracker.

	Pitch		Yaw	
Methods	Avg.	Std.	Avg.	Std.
Funes Mora et al. [26]	**4.17**	**5.59**	6.89	14.42
HPE_GLLiM [10]	7.94	9.23	10.62	11.95
Kalman filter [1, 6]	23.17	18.67	25.55	21.22
HPE_SKF	5.34	8.30	**6.68**	**9.76**

The results obtained with the Biwi-Kinect and EYEDIAP datasets are summarized in Table 1 and Table 2, respectively, namely the average and standard deviations of the absolute error between the head-pose estimated values and the ground-truth values. The proposed tracking method improves head-pose parameter tracking, compared to all the other methods. For example, the average error for the yaw angles in Table 1 is of 8.6° while all other methods yield an error larger than 12°. We observe the same behavior for the pitch angle. Moreover, our method also reduces the standard deviation. Compared to the landmark-based method of [39], our method is able to provide an estimation for each test input, whereas the method based on landmarks is unable to provide an output when some of the landmarks are not visible due to extreme head orientations. In this case [39] yields very large errors, *e.g*. first row of Table 1. We also note that the proposed HPE_SKF method performs much better than a standard Kalman filter. On Table 2 the global average error and standard deviations are lower than Table 1 due to the person-based model learning. The behavior shown in Table 2 follows the same as the one of Table 1, *i.e*. our tracking method decreases the head pose estimation average error and standard deviation with respect to the HPE_GLLiM and the classic Kalman filtering. [26] (that released the EYEDIAP dataset) achieves better results for pitch, while our method obtains better results for yaw.

7. Conclusions

Head-pose trackers have the advantage of combining information from both past and present, and hence they avoid oscillations between consecutive estimations solely based on independent observations, as it is the case with many head-pose estimation methods, e.g. Fig. 2 and Fig. 4. Overall, the output of our tracking method is both more accurate and smoother than the output of several head-pose methods (with and without tracking). Moreover, noisy observations, *e.g*. due to badly aligned bounding boxes or to partial occlusions, do not impact too much the proposed tracker because the temporal model, once properly trained, does not allow oscillations between consecutive estimations. Furthermore, our approach does not only reduce both the estimation error and the standard deviation, it smoothes the estimation of head poses over the whole video. This is very useful for other temporal tasks, such as the estimation of eye gaze or of the visual focus of attention [23].

The method presented in this paper is very general and it is limited neither to head pose nor to HOG features. It can be applied to estimate and track the pose parameters of objects of all kinds, provided that their image appearance varies as a function of their 3D orientation in a consistent way. Therefore, one can combine our tracker with other types of features, such as features obtained by training a neural network and by substituting the last layer of the latter with the generative regression model used above.

References

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

[2] S. O. Ba and J.-M. Odobez. A probabilistic framework for joint head tracking and pose estimation. In *IEEE ICPR*, August 2004.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, June 2008.

[4] C. BenAbdelkader. Robust head pose estimation using supervised manifold learning. In *ECCV*. Springer, 2010.

[5] A. Bhattachayya. On a measure of divergence between two statistical population defined by their population distributions. *Bulletin Calcutta Mathematical Society*, 35:99–109, July 1943.

[6] C. Bishop. Pattern recognition and machine learning, 2007.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE CVPR*, June 2005.

[8] A. Deleforge, F. Forbes, and R. Horaud. High-dimensional regression with Gaussian mixtures and partially-latent response variables. *Statistics and Computing*, 25(5):893–911, September 2015.

[9] F. Dornaika and F. Davoine. Head and facial animation tracking using appearance-adaptive models and particle filters. In *IEEE CVPRW*, June 2004.

[10] V. Drouard, S. Ba, G. Evangelidis, A. Deleforge, and R. Horaud. Head pose estimation via probabilistic high-dimensional regression. In *IEEE ICIP*, September 2015.

[11] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool. Random forests for real time 3d face analysis. *IJCV*, 101(3):437–458, February 2013.

[12] J. Foytik and V. Asari. A two-layer framework for piecewise linear manifold-based head pose estimation. *International Journal of Computer Vision*, 101(2):270–287, January 2013.

[13] A. Gee and R. Cipolla. Fast visual tracking by temporal consensus. *Image and Vision Computing*, 14(2):105–114, March 1996.

[14] Z. Ghahramani and G. E. Hinton. Switching state-space models. Technical report, Citeseer, 1996.

[15] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE CVPR*, 2014.

[16] N. Hu, W. Huang, and S. Ranganath. Head pose estimation by non-linear embedding and mapping. In *IEEE ICIP*, September 2005.

[17] C.-J. Kim. Dynamic linear models with markov-switching. *Journal of Econometrics*, 60(1-2):1–22, January–February 1994.

[18] J. F. Kooij, G. Englebienne, and D. M. Gavrila. A non-parametric hierarchical model to discover behavior dynamics from tracks. In *ECCV*. Springer, 2012.

[19] Z. Li, Y. Fu, J. Yuan, T. Huang, and Y. Wu. Query driven localized linear discriminant models for head pose estimation. In *IEEE International Conference on Multimedia and Expo*, July 2007.

[20] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, November 2004.

[21] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *IEEE ICCV*, December 2015.

[22] M. Marin-Jimenez, A. Zisserman, M. Eichner, and V. Ferrari. Detecting people looking at each other in videos. *International Journal of Computer Vision*, 106(3):282–296, February 2014.

[23] B. Massé, S. Ba, and R. Horaud. Simultaneous estimation of gaze direction and visual focus of attention for multi-person-to-robot interaction. In *IEEE International Conference on Multimedia and Expo*, July 2016.

[24] T. Maurer and C. von der Malsburg. Tracking and learning graphs and pose on image sequences of faces. In *International Conference on Automatic Face and Gesture Recognition*, October 1996.

[25] K. A. F. Mora, F. Monay, and J.-M. Odobez. Eyediap: a database for the development and evaluation of gaze estimation algorithms from rgb and rgb-d cameras. In *ACM Symposium on Eye Tracking Research and Applications*, pages 255–258, March 2014.

[26] K. A. F. Mora and J.-M. Odobez. Gaze estimation from multimodal kinect data. In *IEEE CVPRW*, June 2012.

[27] K. P. Murphy. Switching kalman filters. Technical report, Citeseer, 1998.

[28] E. Murphy-Chutorian, A. Doshi, and M. Trivedi. Head pose estimation for driver assistance systems: A robust algorithm and experimental evaluation. In *IEEE Intelligent Transportation Systems Conference*, Sept 2007.

[29] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation in computer vision: a survey. *IEEE TPAMI*, 31(4):607–626, April 2009.

[30] S. M. Oh, J. M. Rehg, T. Balch, and F. Dellaert. Learning and inference in parametric switching linear dynamic systems. In *IEEE ICCV*, 2005.

[31] V. Pavlovic, J. M. Rehg, and J. MacCormick. Learning switching linear models of human motion. In *Conference on Neural Information Processing Systems*, 2000.

[32] B. Raytchev, I. Yoda, and K. Sakaue. Head pose estimation by nonlinear manifold learning. In *IEEE ICPR*, August 2004.

[33] D. Salmond. Mixture reduction algorithms for point and extended object tracking in clutter. *IEEE Transactions on Aerospace and Electronic Systems*, 45(2):667–686, 2009.

[34] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *IEEE CVPRW*, June 2014.

[35] S. Srinivasan and K. L. Boyer. Head pose estimation using view based eigenspaces. *IEEE ICPR*, June 2002.

[36] K. Sundararajan and D. L. Woodard. Head pose estimation in the wild using approximate view manifolds. In *IEEE CVPRW*, June 2015.

[37] S. Taheri, A. C. Sankaranarayanan, and R. Chellappa. Joint albedo estimation and pose tracking from video. *IEEE TPAMI*, 35(7):1674–1689, July 2013.

[38] J. Tu, T. Huang, and H. Tao. Accurate head pose tracking in low resolution video. In *IEEE International Conference on Automatic Face and Gesture Recognition*, April 2006.

[39] M. Uřičář, V. Franc, and V. Hlaváč. Detector of facial landmarks learned by the structured output SVM. In *International Conference on Computer Vision Theory and Applications*, February 2012.

[40] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE CVPR*, June 2001.

[41] R. Yang and Z. Zhang. Model-based head pose tracking with stereovision. In *IEEE International Conference on Automatic Face and Gesture Recognition*, May 2002.

[42] P. Yao, G. Evans, and A. Calway. Using affine correspondence to estimate 3-d facial pose. In *IEEE ICIP*, October 2001.