Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2008, Article ID 823695, 17 pages
doi:10.1155/2008/823695

Research Article

Simultaneous Eye Tracking and Blink Detection with Interactive Particle Filters

Junwen Wu and Mohan M. Trivedi

Computer Vision and Robotics Research Laboratory, University of California, San Diego, La Jolla, CA 92093, USA

Correspondence should be addressed to Junwen Wu, juwu@ucsd.edu

Received 2 May 2007; Revised 1 October 2007; Accepted 28 October 2007

Recommended by Juwei Lu

We present a system that simultaneously tracks eyes and detects eye blinks. Two interactive particle filters are used for this purpose, one for the closed eyes and the other one for the open eyes. Each particle filter is used to track the eye locations as well as the scales of the eye subjects. The set of particles that gives higher confidence is defined as the primary set and the other one is defined as the secondary set. The eye location is estimated by the primary particle filter, and whether the eye status is open or closed is also decided by the label of the primary particle filter. When a new frame comes, the secondary particle filter is reinitialized according to the estimates from the primary particle filter. We use autoregression models for describing the state transition and a classification-based model for measuring the observation. Tensor subspace analysis is used for feature extraction which is followed by a logistic regression model to give the posterior estimation. The performance is carefully evaluated from two aspects: the blink detection rate and the tracking accuracy. The blink detection rate is evaluated using videos from varying scenarios, and the tracking accuracy is given by comparing with the benchmark data obtained using the Vicon motion capturing system. The setup for obtaining benchmark data for tracking accuracy evaluation is presented and experimental results are shown. Extensive experimental evaluations validate the capability of the algorithm.

1. INTRODUCTION

Eye blink detection plays an important role in human-computer interface (HCI) systems. It can also be used in driver's assistance systems. Studies show that eye blink duration has a close relation to a subject's drowsiness [1]. The openness of eyes, as well as the frequency of eye blinks, shows the level of the person's consciousness, which has potential applications in monitoring driver's vigourous level for additional safety control [2]. Also, eye blinks can be used as a method of communication for people with severe disabilities, in which blink patterns can be interpreted as semiotic messages [3–5]. This provides an alternate input modality to control a computer: communication by "blink pattern." The duration of eye closure determines whether the blink is voluntary or involuntary. Blink patterns are used by interpreting voluntary long blinks according to the predefined semiotics dictionary, while ignoring involuntary short blinks.

Eye blink detection has attracted considerable research interest from the computer vision community. In literature, most existing techniques use two separate steps for eye tracking and blink detection [2, 3, 5–8]. For eye blink detection systems, there are three types of dynamic information involved: the global motion of eyes (which can be used to infer the head motion), the local motion of eye pupils, and the eye openness/closure. Accordingly, an effective eye tracking algorithm for blink detection purposes needs to satisfy the following constraints:

(i) track the global motion of eyes, which is confined by the head motion;
(ii) maintain invariance to local motion of eye pupils;
(iii) classify the closed-eye frames from the open-eye frames.

Once the eyes' locations are estimated by the tracking algorithm, the differences in image appearance between the open eyes and the closed eyes can be used to find the frames in which the subjects' eyes are closed, such that eye blinking can be determined. In [2], template matching is used to track the eyes and color features are used to determine the

openness of eyes. Detected blinks are then used together with pose and gaze estimates to monitor the driver's alertness. In [6, 9], blink detection is implemented as part of a large facial expression classification system. Differences in intensity values between the upper eye and lower eye are used for eye openness/closure classification, such that closed-eye frames can be detected. The use of low-level features makes the real-time implementation of the blink detection systems feasible. However, for videos with large variations, such as the typical videos collected from in-car cameras, the acquired images are usually noisy and with low-resolution. In such scenarios, simple low-level features, like color and image differences, are not sufficient. Temporal information is also used by some other researchers for blinking detection purposes. For example, in [3, 5, 7], the image difference between neighboring frames is used to locate the eyes, and the temporal image correlation is used thereafter to determine whether the eyes are open or closed. This system provides a possible new solution for a human-computer interaction system that can be used for highly disabled people. Besides that, motion information has been exploited as well. The estimate of the dense motion field describes the motion patterns, in which the eye lid movements can be separated to detect eye blinks. In [8], dense optical flow is used for this purpose. The ability to differentiate the motion related to blinks from the global head motion is essential. Since face subjects are nonrigid and nonplanar, it is not a trivial work.

Such two-step-based blink detection system requires that the tracking algorithms are capable of handling the appearance change between the open eyes and the closed eyes. In this work, we propose an alternative way that simultaneously tracks eyes and detects eye blinks. We use two interactive particle filters, one tracks the open eyes and the other one tracks the closed eyes. Eye detection algorithms can be used to give the initial position of the eyes [10–12], and after that the interactive particle filters are used for eye tracking and blink detection. The set of particles that gives higher confidence is defined as the primary particle set and the other one is defined as the secondary particle set. Estimates of the eyes' location, as well as the eye class labels (open-eye versus closed-eye), are determined by the primary particle filter, which is also used to reinitialize the secondary particle filter for the new observation. For each particle filter, the state variables characterize the location and size of the eyes. We use autoregression (AR) models to describe the state transitions, where the location is modeled by a second-order AR and the scale is modeled by a separate first-order AR. The observation model is a classification-based model, which tracks eyes according to the knowledge learned from examples instead of the templates adapted from previous frames. Therefore, it can avoid accumulation of the tracking errors. In our work, we use a regression model in tensor subspace to measure the posterior probabilities of the observations. Other classification/regression models can be used as well. Experimental results show the capability of the algorithm.

The remaining part of the paper is organized as follows. In Section 2, the theoretical foundation of the particle filter is reviewed. In Section 3, the details of the proposed algorithm are presented. The system flowchart in Figure 1 gives an overview of the algorithm. In Section 4, a systematic experimental evaluation of the performance is described. The performance is evaluated from two aspects: the blink detection rate and the tracking accuracy. The blink detection rate is evaluated using videos collected under varying scenarios, and the tracking accuracy is evaluated using benchmark data collected with the Vicon motion capturing system. Section 5 gives some discussion and concludes the paper.

2. DYNAMIC SYSTEMS AND PARTICLE FILTERS

The fundamental prerequisite of a simultaneous eye tracking and blink detection system is to accurately recover the dynamics of eyes, which can be modeled by a dynamic system. Open eyes and closed eyes appear to have significantly different appearances. A straightforward way is to model the dynamics of open-eye and closed-eye individually. We use two interactive particle filters for this purpose. The posterior probabilities learned by the particle filters are used to determine which particle filter gives the correct tracks, and this particle filter is thus labeled as the primary one. Figure 1 gives the diagram of the system. Since the particle filters are the key part of this blink detection system, in this section, we present a detailed overview of the dynamic system and its particle filtering solutions, such that the proposed system for simultaneous eye tracking and blink detection can be better understood.

2.1. Dynamic systems

A dynamic system can be described by two mathematical models. One is the state-transition model, which describes the system evolution rules, represented by the stochastic process $\{\mathbf{S}_t\} \in \mathcal{R}^{n_s \times 1}$ $(t = 0, 1, \dots)$, where

$$\mathbf{S}_t = F_t(\mathbf{S}_{t-1}, \mathbf{V}_t). \tag{1}$$

$\mathbf{V}_t \in \mathcal{R}^{n_v \times 1}$ is the state transition noise with known probability density function (PDF) $p(\mathbf{V}_t)$. The other one is the observation model, which shows the relationship between the observable measurement of the system and the underlying hidden state variables. The dynamic system is observed at discrete times t via realization of the stochastic process, modeled as follows:

$$\mathbf{Y}_t = H_t(\mathbf{S}_t, \mathbf{W}_t). \tag{2}$$

$\mathbf{Y}_t$ $(t = 0, 1, \dots)$ is the discrete observation obtained at time t. $\mathbf{W}_t \in \mathcal{R}^{n_w}$ is the observation noise with known PDF $p(\mathbf{W}_t)$, which is independent from $\mathbf{V}_t$. For simplicity, we use capital letters to refer to the random processes and lowercase letters to denote the realization of the random processes.

Given that these two system models are known, the problem is to estimate any function of the state $f(\mathbf{S}_t)$ using the expectation $\mathrm{E}[f(\mathbf{S}_t) \mid \mathbf{Y}_{0:t}]$. If F_t and H_t are linear, and the two noise PDFs, $p(\mathbf{V}_t)$ and $p(\mathbf{W}_t)$, are Gaussian, the system can be characterized by a Kalman filter [13]. Unfortunately, Kalman filters only provide the first-order approximations for general systems. Extended Kalman Filter (EKF) [13] is one way to handle the nonlinearity. A more general

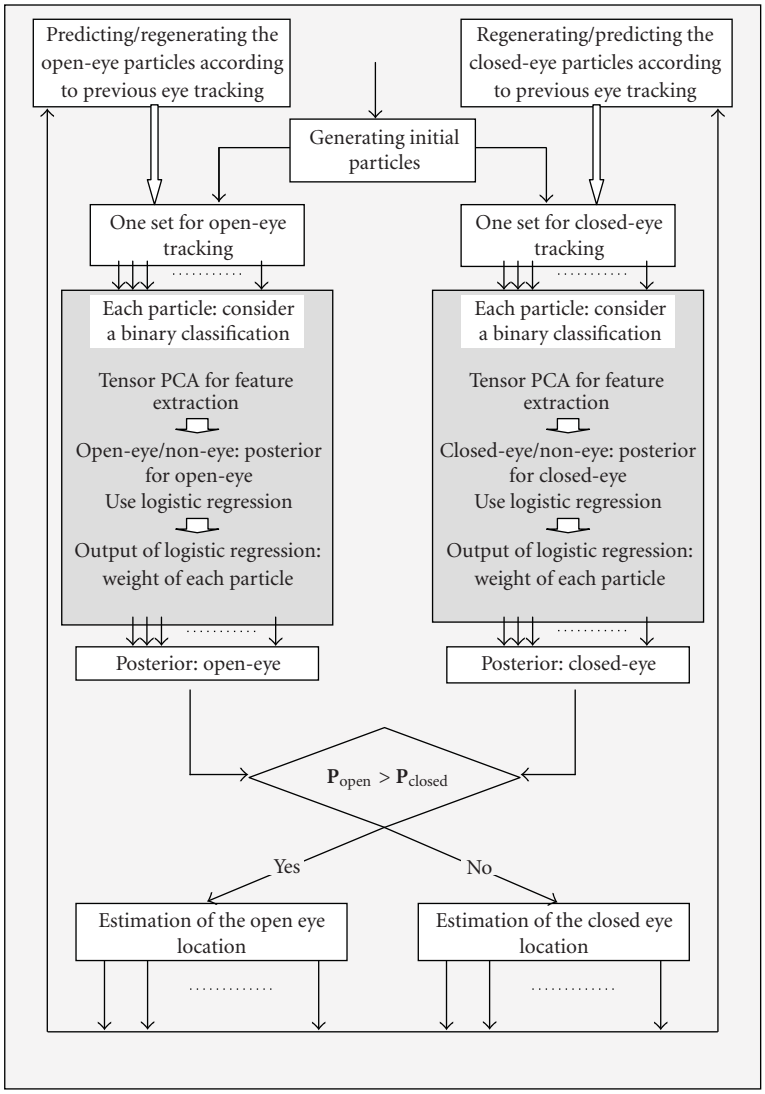


Figure 1: Flow-chart for eye blink detection system. For every new frame observation, new particles are first predicted from the known important distribution, and then updated accordingly based on the posterior estimated by logistic regressor in the tensor subspaces. The best estimation gives the class label (open-eye/closed-eye) as well as the eye location.

framework is provided by particle filtering techniques. Particle filtering is a Monte Carlo solution for general form dynamic systems. As an alternative to the EKF, particle filters have the advantage that with sufficient samples, the solutions approach the Bayesian estimate.

2.2. Review of a basic particle filter

Particle filters are sequential analogues of Markov chain Monte Carlo (MCMC) batch methods. They are also known as sequential Monte Carlo (SMC) methods. Particle filters are widely used in positioning, navigation, and tracking for modeling dynamic systems [14–20]. The basic idea of particle filtering is to use point mass, or particles, to represent the probability densities. The tracking problem can be expressed as a Bayes filtering problem, in which the posterior distribution of the target state is updated recursively as a new observation comes in

$$p(\mathbf{S}_t \mid \mathbf{Y}_{0:t}) \propto p(\mathbf{Y}_t \mid \mathbf{S}_t; \mathbf{Y}_{0:t-1}) \int_{\mathbf{S}_{t-1}} p(\mathbf{S}_t \mid \mathbf{S}_{t-1}; \mathbf{Y}_{0:t-1}) \times p(\mathbf{S}_{t-1} \mid \mathbf{Y}_{0:t-1}) d\mathbf{S}_{t-1}. \tag{3}$$

The likelihood $p(\mathbf{Y}_t \mid \mathbf{S}_t; \mathbf{Y}_{0:t-1})$ is the observation model, and $p(\mathbf{S}_t \mid \mathbf{S}_{t-1}; \mathbf{Y}_{0:t-1})$ is the state transition model.

There are several versions of the particle filters, such as sequential importance sampling (SIS) [21, 22]/sampling-importance resampling (SIR) [22–24], auxiliary particle filters [22, 25], and Rao-Blackwellized particle filters [20, 22, 26, 27], and so forth. All particle filters are derived based on the following two assumptions. The first assumption is that

the state-transition is a first-order Markov process, which simplifies the state transition model in (3) to

$$p(\mathbf{S}_t \mid \mathbf{S}_{t-1}; \mathbf{Y}_{0:t-1}) = p(\mathbf{S}_t \mid \mathbf{S}_{t-1}). \tag{4}$$

The second assumption is that the observations $\mathbf{Y}_{1:t}$ are conditionally independent given known states $\mathbf{S}_{1:t}$, which implies that each observation only relies on the current state; then we have

$$p(\mathbf{Y}_t \mid \mathbf{S}_t; \mathbf{Y}_{0:t-1}) = p(\mathbf{Y}_t \mid \mathbf{S}_t). \tag{5}$$

These two assumptions simplify the Bayes filter in (3) to

$$p(\mathbf{S}_t \mid \mathbf{Y}_{0:t}) \propto p(\mathbf{Y}_t \mid \mathbf{S}_t) \int_{\mathbf{S}_{t-1}} p(\mathbf{S}_t \mid \mathbf{S}_{t-1}) p(\mathbf{S}_{t-1} \mid \mathbf{Y}_{0:t-1}) d\mathbf{S}_{t-1}. \tag{6}$$

Exploiting this, particle filter uses a number of particles $(\omega^{(i)}, \mathbf{s}_t^{(i)})$ to sequentially compute the expectation of any function of the state, which is $\mathrm{E}[f(\mathbf{S}_t) \mid \mathbf{y}_{0:t}]$, by

$$\mathrm{E}[f(\mathbf{S}_t) \mid \mathbf{y}_{0:t}] = \int f(\mathbf{s}_t) p(\mathbf{s}_t \mid \mathbf{y}_{0:t}) d\mathbf{s}_t = \sum_i \omega_t^{(i)} f\left(\mathbf{s}_t^{(i)}\right). \tag{7}$$

In our work, we use the combination of SIS and SIR. Equation (6) tells us that the estimation is achieved by a prediction step, $\int_{\mathbf{s}_{t-1}} p(\mathbf{s}_t \mid \mathbf{s}_{t-1}) p(\mathbf{s}_{t-1} \mid \mathbf{y}_{0:t-1}) d\mathbf{s}_{t-1}$, followed by an update step, $p(\mathbf{y}_t \mid \mathbf{s}_t)$. At the prediction step, the new state $\hat{\boldsymbol{s}}_t^i$ is sampled from the state evolution process $F_{t-1}(\mathbf{s}_{t-1}^{(i)}, \cdot)$ to generate a new cloud of particle filters. With the predicted state $\hat{\boldsymbol{s}}_t^i$, an estimate of the observation is obtained, which is used in the update step to correct the posterior estimate. Each particle is then reweighted in proportion to the likelihood of the observation at time t. We adopt the idea of "resampling when necessary" as suggested by [21, 28, 29], which suggests that resampling is only necessary when the effective number of particles is sufficiently low. The SIS/SIR algorithm can be summarized as in Algorithm 1.

$\pi(\mathbf{s}_t^{(i)} \mid \mathbf{s}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}) = \pi(\mathbf{s}_t^{(i)} \mid \mathbf{s}_{t-1}^{(i)}, \mathbf{y}_{0:t})$ is also called the proposal distribution. A common and simple choice is to use the prior distribution [30] as the proposal distribution, which is also known as a bootstrap filter. We use the bootstrap filter in our work, and by this way the weight update can be simplified to

$$\hat{\omega}_t^{(i)} = \omega_{t-1}^{(i)} p\left(\mathbf{y}_t \mid \mathbf{s}_t^{(i)}\right). \tag{12}$$

This indicates that the weight update is directly related to the observational model.

3. PARTICLE FILTERS FOR EYE TRACKING AND BLINK DETECTION

The appearance of eyes is presented to have significant changes when blinks occur. To effectively handle such appearance changes, we use two interactive particle filters, one for open eyes and the other one for closed eyes. These two particle filters are only different in the observation measurement. In the following sections, we present the three elements of the proposed particle filters: state transition model, observation model, and prediction/update scheme.

(1) For $i = 1, \ldots, N$, draw samples from the importance distributions (prediction step):

$$\mathbf{s}_t^{(i)} \sim \pi(\mathbf{s}_t \mid \mathbf{s}_{0:t-1}, \mathbf{y}_{0:t}); \tag{8}$$

(2) Evaluate the importance weights for every particle up to a normalized constant (update step):

$$\hat{\omega}_t^{(i)} = \omega_{t-1}^{(i)} \frac{p\left(\mathbf{y}_t \mid \mathbf{s}_t^{(i)}\right) p\left(\mathbf{s}_t^{(i)} \mid \mathbf{s}_{t-1}^{(i)}\right)}{\pi\left(\mathbf{s}_t^{(i)} \mid \mathbf{s}_{0:t-1}^{(i)}, \mathbf{y}_{0:t}\right)}; \tag{9}$$

(3) Normalize the importance weights:

$$\omega_t^{(i)} = \frac{\hat{\omega}_t^{(i)}}{\sum_{j=1}^N \hat{\omega}_t^{(j)}}, \quad i = 1, \ldots, N; \tag{10}$$

(4) Compute an estimate of the effective number of the particles:

$$N_{\text{eff}} = \frac{1}{\sum_{i=1}^N \left(\omega_t^{(i)}\right)}; \tag{11}$$

(5) If $N_{\text{eff}} < \theta$, where θ is a given threshold, we perform resampling. N particles are drawn from the current particle set with probabilities proportional to their weights. Replace the current particle set with this new one, and reset each new particle's weight to $1/N$.

Algorithm 1: SIS/SIR particle filter.

3.1. State transition model

The system dynamics, which are described by the state variables, are defined by the location of the eye and the size of the eye image patches. The state vector is $\mathbf{S}_t = (u_t, v_t; \rho_t)$, where (u_t, v_t) defines the location and ρ_t is used to define the size of eye image patches and normalize them to a fixed size. In other words, the state vector $(u_t, v_t; \rho_t)$ means that the image patch under study is centered at (u_t, v_t) and its size is $40\rho_t \times 60\rho_t$, where 40×60 is the fixed size of the eye patches we use in our study.

A second-order autoregressive (AR) model is used for estimating the eyes' movement. The AR model has been widely used in particle filter tracking literature for modeling the motion. It can be written as

$$\begin{aligned} \mathbf{u}_t &= \overline{\boldsymbol{u}} + \mathbf{A}(\mathbf{u}_{t-1} - \overline{\boldsymbol{u}}) + \mathbf{B}\boldsymbol{\mu}_t, \\ \mathbf{v}_t &= \overline{\boldsymbol{v}} + \mathbf{A}(\mathbf{v}_{t-1} - \overline{\boldsymbol{v}}) + \mathbf{B}\boldsymbol{\mu}_t, \end{aligned} \tag{13}$$

where

$$\mathbf{u}_t = \begin{pmatrix} u_t \\ u_{t-1} \end{pmatrix}, \qquad \mathbf{v}_t = \begin{pmatrix} v_t \\ v_{t-1} \end{pmatrix}. \tag{14}$$

$\overline{\boldsymbol{u}}$ and $\overline{\boldsymbol{v}}$ are the corresponding mean values for u and v. As pointed out by [31], this dynamic model is actually a temporal Markov chain. It is capable of capturing complicated

object motion. **A** and **B** are matrices representing the deterministic and the stochastic components, respectively. **A** and **B** can be either obtained by a maximum-likelihood estimation or set manually from prior knowledge. $\boldsymbol{\mu}_t$ is the i.i.d. Gaussian noise.

We use a first-order AR model to model the scale transition, which is

$$\rho_t - \overline{\rho} = C(\rho_{t-1} - \overline{\rho}) + D\eta_t. \tag{15}$$

Similar to the motion model, C is the parameter describing the system deterministic component, and D is the parameter describing the system stochastic component. $\overline{\rho}$ is the mean value of the scales, and η_t is the i.i.d. measurement noise. We assume η_t is uniformly distributed. The scale is crucial for many image appearance-based classifiers. An incorrect scale causes a significant difference in the image appearance. Therefore, the scale transition model is one of the most important prerequisites for obtaining an effective particle filter for measuring the observation. Experimental evaluation shows that the AR model with uniform i.i.d. noise is appropriate for tracking the scale changes.

3.2. Classification-based observation model

In literature, many efforts have been done to address the problem of selecting the proposal distribution [15, 32–35]. A carefully selected proposal distribution can alleviate the sample depletion problem, which refers to the problem that the particle-based posterior approximation collapses over time to a few particles. For example, in [35], AdaBoost is incorporated into the proposal distribution to form a mixture proposal. This is crucial in some typical occlusion scenarios, since "cross over" targets can be represented by the mixture-model. However, the introduction of complicated proposal distributions greatly increases the computational complexity. Also, since blink detection is usually a single-target tracking problem, the proposal distribution is more likely to be single-mode. Therefore, we only use bootstrap particle filtering approach, and avoid the nontrivial proposal distribution estimation problem.

In this work, we focus on a better observation model $p(\mathbf{y}_t \mid \mathbf{s}_t)$. The rationale is based on the observation that combined with the resampling step, a more accurate likelihood learning from a better observation model can move the particles to areas of high likelihood. This will in turn mitigate the sample depletion problem, leading to a significant increase in performance. In literatures, many existing approaches use simple online template matching [16, 18, 19, 36] to get the observation model, where the templates are constructed from low-level features, such as color, edges, contour, and so forth, from previous observations. The likelihood is usually estimated based on a Gaussian distribution assumption [26, 34]. However, such approaches in a large extent rely on a reasonably stable feature detection algorithm. Also, usually a large number of the single low-level feature points are needed. For example, the contour-based method requires that the state vector be able to describe the evolution of all contour points. This results in a high-dimensional state space. Correspondingly, the computational cost is expensive. One solution is to use abstracted statistics of these single feature points, such as using color histogram instead of direct color measurement. However, this causes a loss in the spatial layout information, which implies a sacrifice in the localization accuracy. Instead we use a subspace-based classification model for measuring the observation such that a more accurate probability evaluation can be obtained. Statistics learned from a set of training samples are used for classification instead of simple template matching and online updating. This can greatly alleviate the problem of error accumulation. The likelihood estimation problem, $p(\mathbf{y}_t^{(i)} \mid \mathbf{s}_t^{(i)})$, becomes a problem of estimating the distribution of a Bernoulli variable, which is $p(y_t^{(i)} = 1 \mid \mathbf{s}_t^{(i)})$. $y_t^{(i)} = 1$ means that the current state generates a positive example. In our eye tracking and blink detection problem, it represents that an eye patch is located, including both open eye and closed eye. Logistic regression is a straightforward solution for this purpose. Obviously, other existing classification/regression techniques can be used as well.

Such classification-based particle filtering framework makes simultaneous tracking and recognition feasible and straightforward. There are two different ways to embed the recognition problem. The first approach is to use a single particle filter, whose observation model is a multiclass classifier. The second approach is to use multiple particle filters, where for each particle filter its observation model uses a binary classifier designed for a specific object class. The particle filter who gets the highest posterior is used to determine the class label as well as the object location, and at the next frame $t+1$, the other particle filters are reinitialized accordingly. We use the second approach for simultaneous eye tracking and blink detection. Individual observation models are built for open eye and closed eye separately, such that two interactive sets of particles can be obtained. The observation models contain two parts: tensor subspace analysis for feature extraction, and logistic regression for class posterior learning. The two parts are individually discussed in Sections 3.2.1 and 3.2.2. Posterior probabilities measured by particles from these two particle filters are individually denoted as $p_o = p(y_t = 1_{\text{oe}} \mid \mathbf{s}_t)$ and $p_c = p(y_t = 1_{\text{ce}} \mid \mathbf{s}_t)$, respectively, where $y_t = 1_{\text{oe}}$ refers to the presence of an open eye and $y_t = 1_{\text{ce}}$ refers to the presence of a closed eye.

3.2.1. Subspace analysis for feature extraction

Most existing applications of using particle filters for visual tracking involve high-dimensional observations. With the increase of the dimensionality in observations, the number of particles required increases exponentially. Therefore, lower dimensional feature extraction is necessary. Sparse low-level features, such as the abstracted statistics of the low-level features, have been proposed for this purpose. Examples of the most commonly used features are color histogram [35, 37], edge density [15, 38], salient points [39], contour points [18, 19], and so forth. The use of such features makes the system capable of easily accommodating the scale changes and handling occlusions; however, performance of

such approaches rely on the robustness of the feature detection algorithms. For example, color histogram is widely used for pedestrian and human face tracking; however, its performance suffers from the illumination changes. Also, the spatial information and the texture information are discarded, which may cause the degradation of the localization accuracy and in turn deteriorate the performance of the successive recognition algorithms.

Instead of these variants of low-level features, we use eigen-subspace for feature extraction and dimensionality reduction. Eigenspace projection provides a holistic feature representation that preserves spatial and textural information. It has been widely exploited in computer vision applications. For example, eigenface has been an effective face recognition technique for decades. Eigenface focuses on finding the most representative lower-dimensional space in which the pattern of the input can be best described. It tries to find a set of "standardized face ingredients" learned from a set of given face samples. Any face image can be decomposed as the combination of these standard faces. However, this principal component analysis- (PCA-) based technique treats each image input as a vector, which causes the ambiguity in image local structure.

Instead of PCA, in [40], a natural alternative for PCA in image domain is proposed, which is the multilinear analysis. Multilinear analysis offers a potent mathematical framework for analyzing the multifactor structure of the image ensemble. For example, a face image ensemble can be analyzed from the following perspectives: identities, head poses, illumination variations, and facial expressions. Multilinear analysis uses tensor algebra to tackle the problem of disentangling these constituent factors. By this way, the sample structures can be better explored and a more informative data representation can be achieved. Under different optimization criterion, variants of the multilinear analysis technique have been proposed. One solution is the direct expansion of the PCA algorithm, TensorPCA from [41], which is obtained under the criteria of the least reconstruction error. Both PCA and tensorPCA are unsupervised techniques, where the class labels are not incorporated in such representations. Here we use a supervised version of the tensor analysis algorithm, which is called tensor subspace analysis (TSA) [42]. Extended from locality preservation projections (LPP) [43], TSA detects the intrinsic geometric structure of the tensor space by learning a lower-dimensional tensor subspace. We compare both observation models of using tensorPCA and TSA. TSA preserves the local structure in the tensor space manifold, hence a better performance should be obtained. Experimental evaluation validates this conjecture. In the following paragraphs, a brief review of the theoretical fundamentals of tensorPCA and TSA are presented.

PCA is a widely used method for dimensionality reduction. PCA offers a well-defined model, which aims to find the subspace that describes the direction of the most variance and at the same time suppress known noise as well as possible. Tensor space analysis is used as a natural alternative for PCA in image domain for efficient computation as well as avoiding ambiguities in image local spatial structure. Tensor space analysis handles images using its natural 2D matrix representation. TensorPCA subspace analysis projects a high-dimensional rank-2 tensor onto a low-dimensional rank-2 tensor space, where the tensor subspace projection minimizes the reconstruction error. Different from the traditional PCA, tensor space analysis provides techniques for decomposing the ensemble in order to disentangle the constituent factors or modes. Since the spatial location is determined by two modes: horizontal position and vertical position, tensor space analysis has the ability to preserve the spatial location, while the dimension of the parameter space is much smaller.

Similarly as the traditional PCA, the tensorPCA projection finds a set of orthogonal bases that information is best preserved. Also, tensorPCA subspace projection decreases the correlation between pixels while the projected coefficient indicates the information preserved on the corresponding tensor basis. However, for tensorPCA, the set of bases are composed by second-order tensors instead of vectors. If we use matrix $\mathbf{X}_i \in \mathcal{R}^{M_1 \times M_2}$ to denote the original image samples, and use matrix $\mathbf{Z}_i \in \mathcal{R}^{P_1 \times P_2}$ as the tensorPCA projection result, tensorPCA can be simply computed by [41]

$$\mathbf{Z}_i = \check{\boldsymbol{U}}^{\mathrm{T}} \mathbf{X}_i \check{\boldsymbol{V}}. \tag{16}$$

The column vectors of the left and right projection matrices $\check{\boldsymbol{U}}$ and $\check{\boldsymbol{V}}$ are the eigenvectors of matrix

$$\mathbf{S}_U = \sum_{i=1}^{N} \left((\mathbf{X}_i - \overline{\mathbf{X}}_m) (\mathbf{X}_i - \overline{\mathbf{X}}_m)^{\mathrm{T}} \right) \tag{17}$$

and matrix

$$\mathbf{S}_V = \sum_{i=1}^{N} \left((\mathbf{X}_i - \overline{\mathbf{X}}_m)^{\mathrm{T}} (\mathbf{X}_i - \overline{\mathbf{X}}_m) \right), \tag{18}$$

respectively; while $\overline{\mathbf{X}}_m = (1/N) \sum_{i=1}^{N} \mathbf{X}_i$. The dimensionality of $\mathbf{Z}_i$ reflects the information preserved, which can be controlled by a parameter α. For example, assume the left projection matrix is computed from $\mathbf{S}_U = \check{\mathbf{U}} \boldsymbol{C} \check{\boldsymbol{U}}^{\mathrm{T}}$, then the rank of the left projection matrix $\check{\mathbf{U}}$ is determined by

$$P_1 = \arg \min_{q} \left\{ \frac{\sum_{i=1}^{q} C_i}{\sum_{i=1}^{M_1} C_i} > \alpha \right\}, \tag{19}$$

where C_i is the ith diagonal element of the diagonal eigenvalue matrix $\mathbf{C}$ ($C_i > C_j$ if $i > j$). The rank of the right projection matrix $\check{\mathbf{V}}$, P_2 can be decided similarly.

TensorPCA is an unsupervised technique. It is not clear whether the information preserved is optimal for classification. Also, only the Euclidean structure is explored instead of the possible underlying nonlinear local structure of the manifold. The Laplacian-based dimensionality reduction technique is an alternate way which focuses on discovering the nonlinear structure of the manifold [44]. It considers preserving the manifold nature while extracting the subspaces. By introducing this idea into tensor space analysis, the following objective function can be obtained [42]:

$$\min_{\mathbf{U},\mathbf{V}} \sum_{i,j} ||\mathbf{U}^{\mathrm{T}} \mathbf{X}_i \mathbf{V} - \mathbf{U}^{\mathrm{T}} \mathbf{X}_j \mathbf{V}|| \mathcal{D}_{i,j}, \tag{20}$$

where $\mathcal{D}_{i,j}$ is the weight matrix of a nearest neighbor graph similar to the one used in LPP [43]:

$$\mathcal{D}_{i,j} = \begin{cases} \exp\left\{ -\dfrac{\left(\mathbf{X}_i/||\mathbf{X}_i|| - \mathbf{X}_j/||\mathbf{X}_j||\right)^2}{2} \right\} \\ \quad \text{if } \mathbf{X}_i \text{ and } \mathbf{X}_j \text{ are from the same class,} \\ 0 \quad \text{if } \mathbf{X}_i \text{ and } \mathbf{X}_j \text{ are from different classes.} \end{cases} \tag{21}$$

We use the iterative approach provided in [42] to compute the left and right projection matrices $\check{\mathbf{U}}$ and $\check{\mathbf{V}}$. The same as tensorPCA, for a given example $\mathbf{X}$, TSA gives

$$\mathbf{Z}_i = \check{\mathbf{U}}^{\mathrm{T}} \mathbf{X}_i \check{\mathbf{V}}. \tag{22}$$

At each frame t, the ith particle determines an observation $\mathbf{X}_t^{(i)}$ from its state $(u_t^{(i)}, v_t^{(i)}; \rho_t^{(i)})$. Tensor analysis extracts the corresponding features $\mathbf{Z}_t^{(i)}$. Now the observation model becomes computing the posterior $p(y_t^{(i)} = 1 \mid \mathbf{Z}_t^{(i)})$. For simplicity, in the following section, we omit the time index t and denote the problem as $p(y^{(i)} = 1 \mid \mathbf{Z}^{(i)})$. Logistic regression is a natural solution for this purpose, which is a generalized linear model for describing the probability of a Bernoulli distributed variable.

3.2.2. Logistic regression for modeling probability

Regression is the problem of modeling the conditional expected value of one random variable based on the observations of some other random variables, which are usually referred to as dependent variables. The variable to model is called the response variable. In the proposed algorithm, the dependent variables are the coefficients from the tensor subspace projection: $\mathbf{Z}^{(i)} = (z_1^{(i)}, \ldots, z_k^{(i)}, \ldots)$, and the response variable to model is the class label $y^{(i)}$, which is a Bernoulli variable that defines the presence of an eye subject. For closed-eye particle filter, this Bernoulli variable defines the presence of a closed eye; while for open-eye particle filter, this variable defines the presence of an open eye.

The relationship between the class label $y^{(i)}$ and its dependent variables, which is the tensor subspace coefficients $(z_1^{(i)}, \ldots, z_k^{(i)}, \ldots)$ here, can be written as

$$y^{(i)} = g\left(\beta_0 + \sum_k \beta_k z_k^{(i)}\right) + e, \tag{23}$$

where e is the error and $g^{-1}(\bullet)$ is called the link function. The variable $y^{(i)}$ can be estimated by

$$E\left(y^{(i)}\right) = g\left(\beta_0 + \sum_k \beta_k z_k^{(i)}\right). \tag{24}$$

Logistic regression uses the *logit* as the link function, which is $\mathrm{logit}(p) = \log(p/(1-p))$. Therefore, the probability of the presence of an eye subject can be modeled as

$$p\left(y^{(i)} = 1 \mid \mathbf{Z}^{(i)}\right) = \frac{\exp^{\beta_0 + \sum_k \beta_k z_k^{(i)}}}{1 + \exp^{\beta_0 + \sum_k \beta_k z_k^{(i)}}}, \tag{25}$$

where $y^{(i)} = 1$ means that an eye subject is present.

3.3. State update

The observation models for open eye and closed eye are individually trained. We have one TSA subspace learned from open eye/noneye training samples, and another TSA subspace learned from closed eye/noneye training samples. Each TSA projection determines a set of transformed features, which are denoted as $\{\mathbf{Z}_{\mathrm{oe}}^{(i)}\}$ and $\{\mathbf{Z}_{\mathrm{ce}}^{(i)}\}$. $\mathbf{Z}_{\mathrm{oe}}^{(i)}$ is the trans-formed TSA coefficients for the open eyes and $\mathbf{Z}_{\mathrm{ce}}^{(i)}$ is the transformed TSA coefficients for the closed eyes. Correspondingly, for open eye and closed eye, individual logistic regression models are used separately for modeling p_c and p_o as follows:

$$p_o^{(i)} = p_{\mathrm{oe}}\left(y^{(i)} = 1 \mid \mathbf{Z}_{\mathrm{oe}}^{(i)}\right), \qquad p_c^{(i)} = p_{\mathrm{ce}}\left(y^{(i)} = 1 \mid \mathbf{Z}_{\mathrm{ce}}^{(i)}\right). \tag{26}$$

The posteriors are used to update the weights of the corresponding particles, as indicated in (12). The updated weights are $\omega_c^{(i)}$ and $\omega_o^{(i)}$.

If we have

$$\max_i p_o^{(i)} > \max_i p_c^{(i)}, \tag{27}$$

it indicates the presence of open eyes, and the particle filter for tracking the open eye is the primary particle filter. Otherwise the eyes of the human subject in the current frame are closed, which indicates the presence of a blink, and the particle filter for the closed eye is determined as the primary particle filter. The use of the max function indicates that our criteria is to trust the most reliable particle. Other criteria can also be used, such as the mean or product of the posteriors from the best K ($K > 1$) particles. The guideline to select the suitable criteria is that only the good particles, which are the particles that reliably indicate the presence of eyes, should be considered. At frame t, assume the particles for the primary particle filter are $\{(u_t^{(i)}, v_t^{(i)}; \rho_t^{(i)}; \omega_t^{(i)})\}$, then the location (u_t, v_t) of the detected eye is determined by

$$u_t = \sum_i \omega_t^{(i)} u_{t-1}^{(i)}; \qquad v_t = \sum_i \omega_t^{(i)} v_{t-1}^{(i)}; \tag{28}$$

and the scale ρ_t of the eye image patch is

$$\rho_t = \sum_i \omega_t^{(i)} \rho_{t-1}^{(i)}. \tag{29}$$

We compute the effective number of particles N_{eff}. If $N_{\mathrm{eff}} < \theta$, we perform resampling for the primary particle filter. The particles with high posteriors are multiplied in proposition to their posteriors. The secondary particle filter is reinitialized by setting the particles' previous states to (u_t, v_t, ρ_t) and the importance weights $\omega_t^{(i)}$ to uniform.

4. EXPERIMENTAL EVALUATION

The performance is evaluated from two aspects: the blink detection accuracy and the tracking accuracy. There are two factors that explain the blink detection rate: first, how many

(a) Frame 94 (miss) (b) Frame 379 (c) Frame 392 (d) Frame 407 (e) Frame 475

Figure 2: Examples of the blink detection results for indoor videos. Red boxes are tracked eyes, and the blue dots are the center of the eye locations. The red bar on the top-left indicates the presence of closed eyes.

(a) Frame 2 (b) Frame 18 (c) Frame 38 (d) Frame 45 (false) (e) Frame 135

Figure 3: Examples of the blink detection results for indoor videos. Red boxes are tracked eyes, and the blue dots are the center of the eye locations. The red bar on the top-left indicates the presence of closed eyes.

blinks are correctly detected; second, the detection accuracy of the blink duration. Videos collected under different scenarios are studied, including indoor videos, in-car videos, and news report videos. A quantitative comparison is listed. To evaluate the tracking accuracy, a benchmark data is required to provide the ground-truth of the eye locations. We use a marker-based motion capturing system to collect the ground-truth data. The experimental setup for obtaining the benchmark data is explained, and the tracking accuracy is presented. Two hundred particles are used for each particle filter if not stated otherwise. For training the tensor subspaces and the logistic regression-based posterior estimators, we use eye samples from FERET gray database to collect open-eye samples. Closed-eye samples are from these three sources: (1) FERET database; (2) Cohn-Kanade AU-coded facial expression database; and (3) online images with closed eye. Noneye samples are from both the FERET database and the online images. We have 273 open-eye images; 149 closed-eye images, and 1879 noneye images. All open-eye, closed-eye, and noneye samples are resized to 40×60 for computing the tensor subspaces and then getting the logistic regressors. With the information-preservation threshold set as $\alpha = 0.9$, the sizes of the tensorPCA subspaces used for modeling the open-eye/noneye and closed-eye/noneye samples are 17×23 and 15×21, respectively; and the sizes of the TSA subspaces for open eye/noneye and closed eye/noneye are 18×22 and 17×22, respectively.

4.1. Blink detection accuracy

We use videos collected under different scenarios for evaluating the blink detection accuracy. In the first set of experiments, we use the videos collected from an indoor lab setting. The subjects are asked to make voluntary long blinks or involuntary short blinks. In the second set of experiments, the videos collected for drivers in outdoor driving scenarios are used. In the third set of experiments, we collect videos for different archormen/women from news reports. In the second and the third experiments, the subjects make natural actions, such as speaking, so only involuntary short blinks are present. We have 8 videos from indoor lab settings; 4 videos of the drivers from an in-car camera; and 20 news report videos, altogether 637 blinks are present. For in-door videos, the frame rate is around 25 frames per second, and each voluntary blink may last 5-6 frames. For in-car videos, the image quality is low, and there are significant illumination changes. Also, the frame rate is fairly low (around 10 frames per second). The voluntary blinks may last around 2-3 frames. For the news report videos, the frame rate is around 15 frames per second. The videos are compressed and the voluntary blinks last for about 3-4 frames. In Table 1. the comparison results are summarized. The true number of blinks, the detected number of blinks, and the number of false positives are shown. Images in Figures 2–8 give some examples of the detection results, which also show the typical video frames we used for studying. Red boxes show the tracked eye location, while blue dots show the center of the tracking results. If there is a red bar on the top right corner, it means that the eyes are closed in the current frame. Examples of the typical false detections or misdetections are also shown.

Table 1

	No. of videos	No. of blinks	No. of correct detections	No. of false positives
Indoor videos	8	133	113	12
In-car videos	4	48	38	11
News report videos	20	456	407	11
Total	32	637	558	34

(a) Frame 4 (b) Frame 35 (c) Frame 108 (false) (d) Frame 127 (e) Frame 210

Figure 4: Examples of the blink detection results for in-car videos. Red boxes are tracked eyes, and the blue dots are the center of the eye locations. The red bar on the top-left indicates the presence of closed eyes.

(a) Frame 42 (b) Frame 302 (false) (c) Frame 349 (d) Frame 489 (e) Frame 769

Figure 5: Examples of the blink detection results for in-car videos. Red boxes are tracked eyes, and the blue dots are the center of the eye locations. The red bar on the top-left indicates the presence of closed eyes.

Blink duration time plays an important role in HCI systems. Involuntary blinks are usually fast while voluntary blinks usually last longer [45]. Therefore, it is also necessary to compare the detected blink duration with the manually labeled true blink duration (in terms of the frame numbers). In Figure 9, we show the detected blink duration in comparison with the manually labeled blink duration. The horizontal axis is the blink index, and the vertical axis shows the duration time in terms of the frame numbers. Experimental evaluation shows that the proposed algorithm is capable of capturing short blinks as well as the long voluntary blinks accurately.

As indicated in (27), the ratio of the posterior maxima, which is $(\max_i p_o^{(i)}/\max_i p_c^{(i)})$, determines the presence of an open eye or close eye. Figure 10(a) shows an example of the obtained ratios for one sequence. Log-scale is used. Let $p_o = \max_i p_o^{(i)}$ and $p_c = \max_i p_c^{(i)}$, the presence of the closed-eye frame is determined when $p_o < p_c$, which corresponds to $\log(p_o/p_c) < 0$ in the log-scale. Examples of the corresponding frames are also shown in Figures 10(b)–10(d) for illustration.

4.2. Comparison of using tensorPCA subspace and TSA subspace

As stated above, by introducing multilinear analysis, the images can better preserve the local spatial structure. However, variants of the tensor subspace basis can be obtained based on different objective functions. TensorPCA is a straightforward extension of the 1D PCA analysis. Both are unsupervised approaches. TSA extends LPP that preserves the nonlinear locality in the manifold, which also incorporates the class information. It is believed that by introducing the local manifold structure and the class information, TSA can obtain a better performance. Experimental evaluations verified this claim. Particle filters that individually use tensorPCA subspace and TSA subspace for observation models are compared for eye tracking and blink detection purpose. Examples of the comparison are shown in Figure 11. As suggested, TSA presents a more accurate tracking result. In Figure 11, examples of the tracking results from both the tensorPCA observation model and the TSA observation model are shown. In each subfigure, the left image shows result from the use of TSA subspace, and the right image shows result from the use of tensorPCA subspace. Just as above, red bounding boxes show the tracked eyes, the blue dots show the center of the detection, and the red bar at the top-right corner indicates the presence of a detected closed-eye frame. For subspace-based analysis, image alignment is critical for classification accuracy. An inaccurate observation model causes errors in the posterior probability computation, which in turn results in inaccurate tracking and poor blink detection.

4.3. Comparison of different scale transition models

It is worth noting that for subspace-based observation model, the scale for normalizing the size of the images is crucial. A bad scale transition model can severely deteriorate the performance. Two different popular models have been used to model the scale transition, and the performance is compared. The first one is the AR model as in (15), and the other one is a Gaussian transition model in which the transition is controlled by a Gaussian distributed random noise, as follows:

$$\rho_t \sim \mathcal{N}(\rho_{t-1}, \sigma^2), \tag{30}$$

where $\mathcal{N}(\rho, \sigma^2)$ is a Gaussian distribution with ρ as the mean and σ^2 as the variance. Examples are shown in Figure 12. The parameters of the Gaussian transition model is obtained by the MAP criteria according to a manually labeled training sequence. In each subfigure, the left image shows the result from using the AR model for scale transition, and the

(a) Frame 10 (b) Frame 141 (c) Frame 230

(d) Frame 269 (e) Frame 300 (f) Frame 370

Figure 6: Examples of the blink detection results for news report videos. Red boxes are tracked eyes, and the blue dots are the center of the eye locations. The red bar on the top-left indicates the presence of closed eyes.

(a) Frame 10 (b) Frame 100 (c) Frame 129

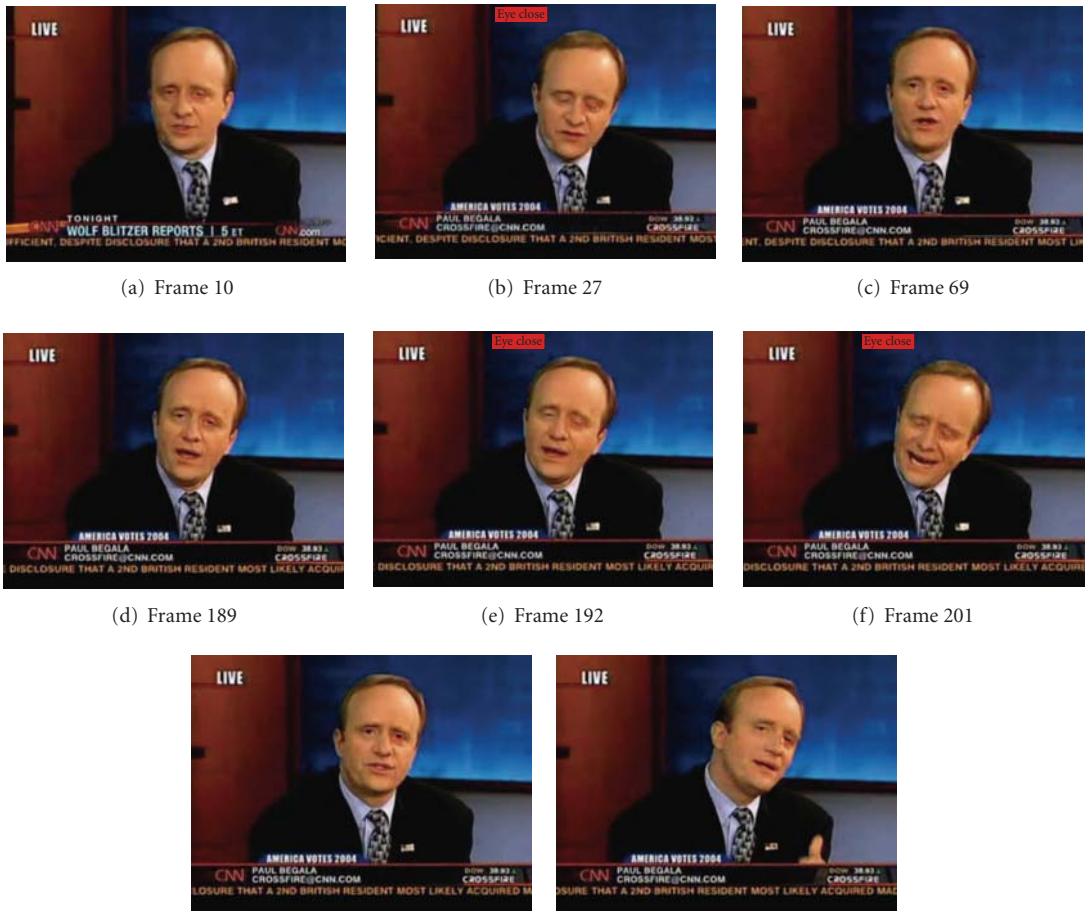
(d) Frame 195 (e) Frame 221 (f) Frame 234 (miss)

Figure 7: Examples of the blink detection results for news report videos. Red boxes are tracked eyes, and the blue dots are the center of the eye locations. The red bar on the top-left indicates the presence of closed eyes.

right one shows the result from using the Gaussian transition model. Experimental results show that AR model performs better. It is because AR model has certain "memory" of the past system dynamics, while Gaussian transition model can only remember the history of its immediate past. Therefore, the "short-memory" of Gaussian transition model uses less information to predict the scale transition trajectory, which is not effective and in turn causes the failure of the tracking.

4.4. *Eye tracking accuracy*

Benchmark data is required for evaluating the tracking accuracy. We use the marker-based Vicon motion capture and analysis system for providing the groundtruth. Vicon system has both hardware and software components. The hardware includes a set of infrared cameras (usually at least 4), controlling hardware modules and a host computer to run the

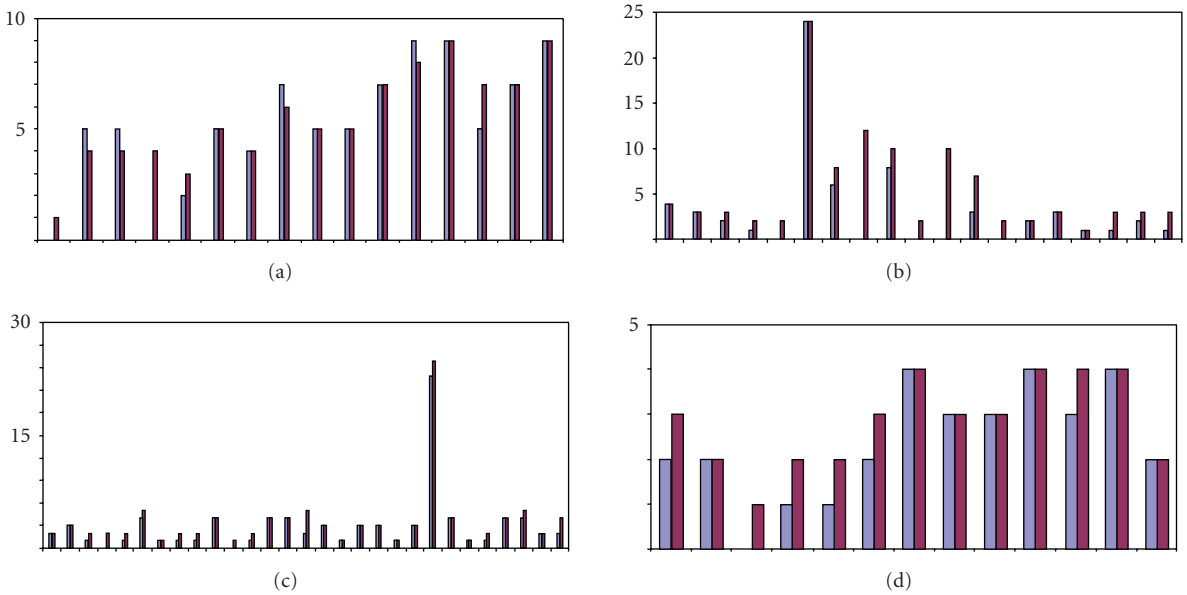


(a) Frame 10 (b) Frame 27 (c) Frame 69

(d) Frame 189 (e) Frame 192 (f) Frame 201

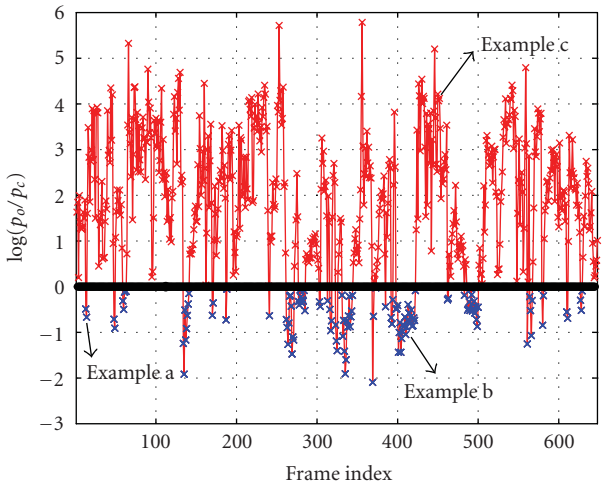
(g) Frame 246 (h) Frame 367

Figure 8: Examples of the blink detection results for news report videos. Red boxes are tracked eyes, and the blue dots are the center of the eye locations. The red bar on the top-left indicates the presence of closed eyes.

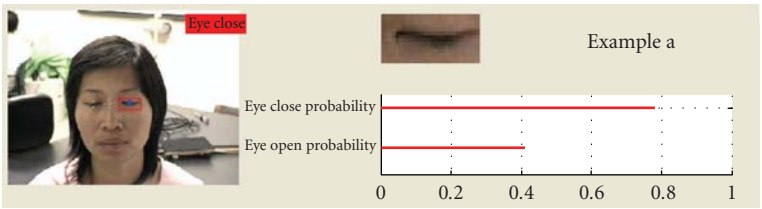


(a) (b) (c) (d)

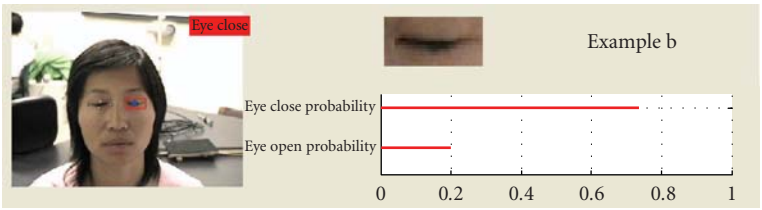
Figure 9: Examples of the duration time of each blink: true blink duration versus detected blink duration. The heights of the bars show the blink duration (in terms of frame numbers). In each pair of bars, the left (blue) bar shows the duration of the detected blink, and the right bar (magenta) shows that of the true blink.



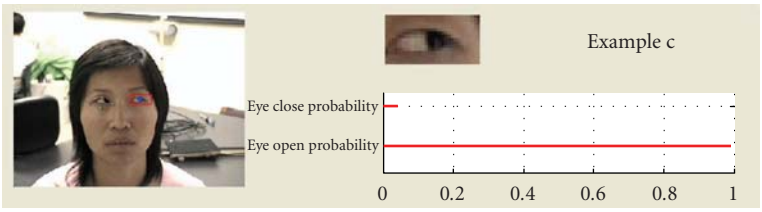
(a) Log ratio of the posteriors of being open-eye (p_o) versus being closed-eye (p_c). Red crosses indicate the open-eye frames, and the blue crosses indicate the detected closed-eye frames



(b)



(c)

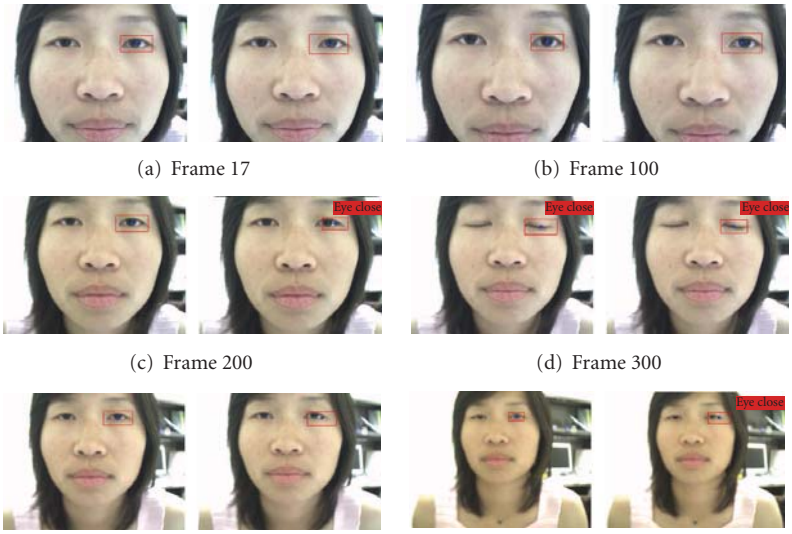


(d)

Figure 10: (a) The log ratio of posteriors $\log(p_o/p_c)$ for each frame in Seq. 5. (b)–(d) The frames corresponding to examples a, b, and c in Figure 10(a). The tracked eyes and the posteriors p_c and p_o are also shown. In each figure, the top red line shows the posterior of being closed eye, and the bottom red line shows the posterior of being open eye.

software. The software includes Vicon IQ that manages, sets up, captures, and processes the motion data, the database manager for keeping records of the data files, their calibration files and the models. We use four Vicon MCAM cameras to track four reflective markers. The setup is shown as in Figure 13. Vicon system tracks the markers' position in Vicon's reference coordinate system, and the video camera collects the video we need for evaluating the proposed algorithm.

Before collecting data, Vicon system requires preprocesses including camera calibration, data acquisition, and model building. With the included calibration tool for the motion capture system, a reflectance marker's 3D position can be obtained in either the Vicon camera coordinate system

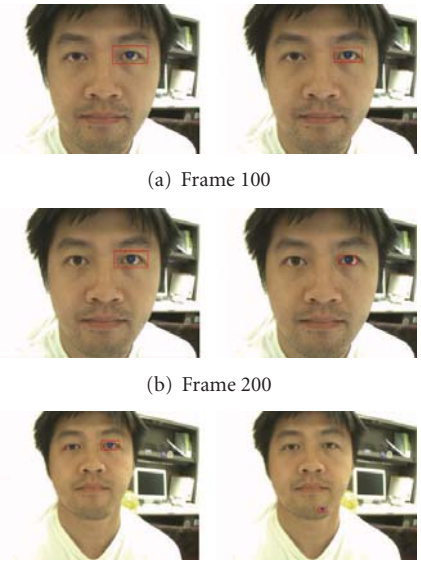


(a) Frame 17 (b) Frame 100

(c) Frame 200 (d) Frame 300

(e) Frame 400 (f) Frame 417

Figure 11: Comparison of using TSA subspace versus using tensorPCA subspace in observation models. In each subfigure, the left image shows the result from using TSA subspace, and the right one shows the result from using tensorPCA subspace.



(a) Frame 100

(b) Frame 200

(c) Frame 380

Figure 12: Comparison of using AR versus using Gaussian transition model in the scale model. In each subfigure, the left image shows the result from AR scale transition model, and the right one shows the result from the Gaussian scale transition model.

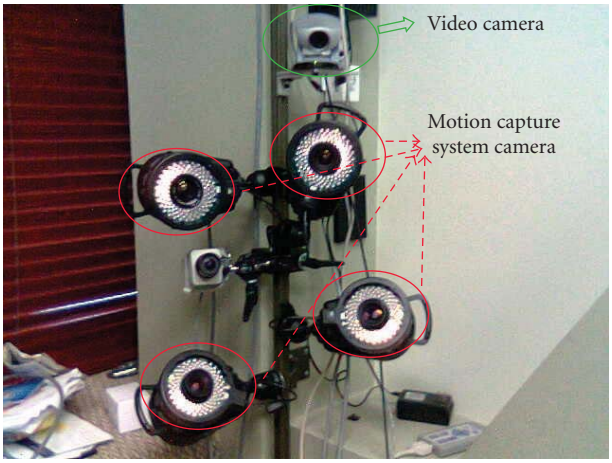


Figure 13: Setup for collecting groundtruth data with Vicon system. Cameras in red circles are Vicon infrared cameras, and the camera in green circle is the video camera for collecting testing sequences.

or an assigned world coordinate system. Since the Vicon camera coordinate system is different from the video camera coordinate system, a calibration between these two camera systems is also required. We use a checker-board pattern with reflectance markers on specified location for this purpose, as shown in Figure 14. Intrinsic parameters **KK** and extrinsic parameters $\mathbf{R}_e$ and $\mathbf{T}_e$ are computed. Intrinsic parameters give the transform from the 3D coordinates in the camera reference frame to the 2D coordinates in the image domain, while extrinsic parameters define the transform between the grid reference frame (as shown in Figure 15) and the camera reference frame. From intrinsic parameters, the 3D coordinates in the camera coordinate system $(X_c, Y_c, Z_c)^{\mathrm{T}}$ can be related with the 2D coordinates in the image plane $(x_p, y_p)^{\mathrm{T}}$ by

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \mathbf{KK}\phi\left(\begin{bmatrix} X_c/Z_c \\ Y_c/Z_c \end{bmatrix}\right), \tag{31}$$

where $\phi(\bullet)$ is a nonlinear function describing the lens distortion. Extrinsic parameters describe the relation between the 3D coordinate in the camera system $\mathbf{M}_c = (X_c, Y_c, Z_c)^{\mathrm{T}}$

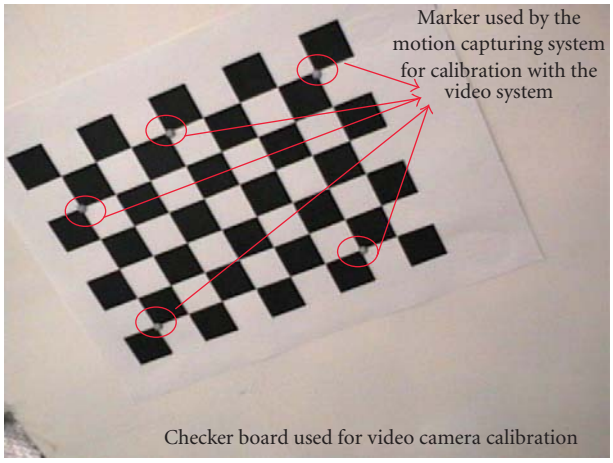


Figure 14: Checker board pattern for calibration between video camera coordinate system and Vicon camera coordinate system. Reflectance markers are put at specific locations.

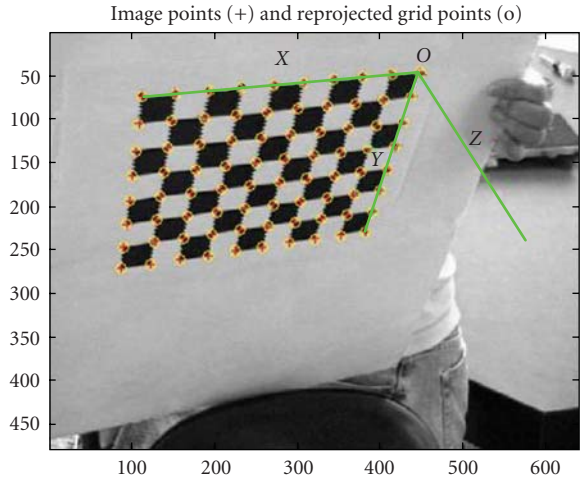


Figure 15: Example of the grid reference frame.

and the 3D coordinate in a given grid reference frame $\mathbf{M}_e = (X_e, Y_e, Z_e)^{\mathrm{T}}$, as follows:

$$\mathbf{M}_c = \mathbf{R}_e \times \mathbf{M}_e + \mathbf{T}_e. \tag{32}$$

Figure 15 gives an example of the grid reference frame. Each pose of the checker-board defines one grid reference frame, hence an individual set of extrinsic parameters can be determined. The reflectance markers are assumed to be infinitely thin, such that their depth can be neglected. Therefore, the reflectance markers' coordinates in current grid reference frame are known, denoted as $\mathbf{M}_e^i$. $\mathbf{M}_e^i$ can be transformed back to the video camera reference frame, which gives the 3D coordinates in the video camera reference frame $\mathbf{M}_c^i$, using the corresponding extrinsic parameters $\mathbf{R}_e^i$ and $\mathbf{T}_e^i$. These markers are also visible by the Vicon system, as shown in Figure 16. Calibrated Vicon system gives the 3D positions of the markers, which are denoted as $\mathbf{M}_v^i$, in the Vicon camera system reference frame. Hence, $\mathbf{M}_c^i$ and $\mathbf{M}_v^i$ can be related by an affine transform:

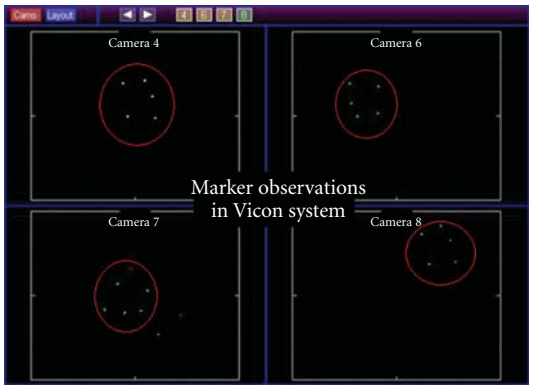


Figure 16: Reflectance markers observed by Vicon IQ system.

$$\mathbf{M}_c^i = \mathbf{R}_{\mathrm{vc}} \times \mathbf{M}_v^i + \mathbf{T}_{\mathrm{vc}}. \tag{33}$$

This relation keeps unchanged when the pose of the checkboard changes. A set of $\{(\mathbf{M}_c^i, \mathbf{M}_e^i)\}$ $(i = 1, \ldots, q)$ can be used to determine this transform. We use the approach proposed by Goryn and Hein in [49] to estimate $\mathbf{R}_{\mathrm{vc}}$ and $\mathbf{T}_{\mathrm{vc}}$. The rotation matrix $\mathbf{R}_{\mathrm{vc}}$ can be determined by least-square approach as follows:

$$\mathbf{R}_{\mathrm{vc}} = \mathbf{W}\mathbf{Q}^{\mathrm{T}}, \tag{34}$$

where $\mathbf{W}$ and $\mathbf{Q}$ are unitary matrices obtained from SVD decomposition of the matrices

$$\begin{aligned} \mathbf{c} &= \frac{1}{N}\sum_{i=1}^{q}\left(\mathbf{M}_c^i - \overline{\boldsymbol{M}}_c\right)\left(\mathbf{M}_v^i - \overline{\boldsymbol{M}}_v\right)^{\mathrm{T}}, \\ \overline{\boldsymbol{M}}_c &= \frac{1}{q}\sum_{i=1}^{q}\mathbf{M}_c^i, \qquad \overline{\boldsymbol{M}}_v = \frac{1}{q}\sum_{i=1}^{q}\mathbf{M}_v^i. \end{aligned} \tag{35}$$

The translation vector $\mathbf{T}_{\mathrm{vc}}$ can be obtained accordingly by

$$\mathbf{T}_{\mathrm{vc}} = \overline{\boldsymbol{M}}_c - \mathbf{R}_{\mathrm{vc}} \times \overline{\boldsymbol{M}}_v. \tag{36}$$

Equation (36) together with (31) determines the mapping from the markers' 3D position given by Vicon system to the 2D pixel position in the image plane. Therefore, with the ViconIQ system providing the markers' 3D positions in Vicon camera systems, we can get our ground-truth data. For reliable tracking, four markers are used, as shown in Figure 17. We use the Vicon system to track the right-eye location as well as providing the scale of the image, and apply the proposed algorithm on tracking and blink detection of left eye. After normalization with the scale, the distance between the right eye and left eye is constant, so that the benchmark data can be used for evaluating the tracking accuracy. The fixed size for computing the subspace is 40×60. We use the center of the markers as the groundtruth for eyes' location.

Figure 18 gives an example of the tracking accuracy. The horizontal axis shows the frame number, and the vertical axis

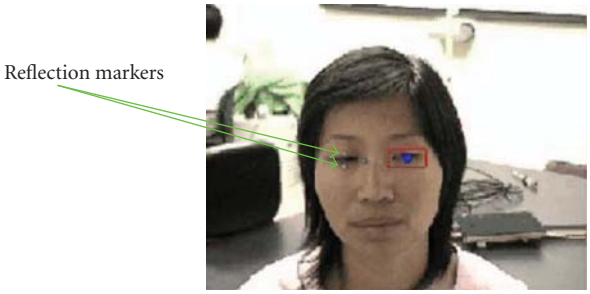


Figure 17: Marker deployment for tracking accuracy benchmark data collection.

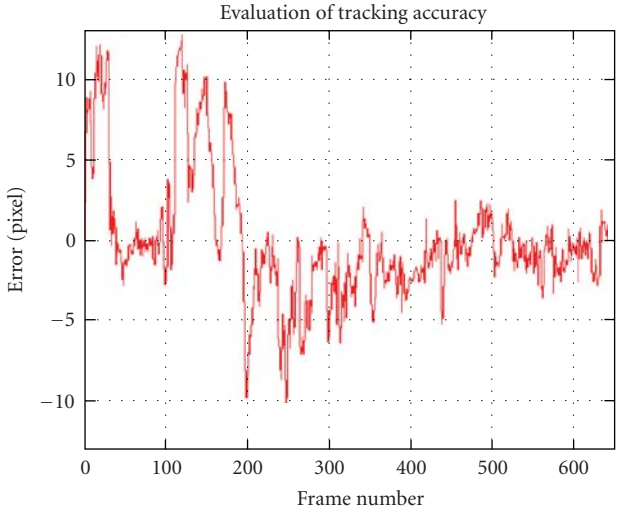


Figure 18: Tracking error after normalization using the scales. The horizontal axis is the frame index, and the vertical axis is the tracking error in pixels after normalization with the scales.

shows the error in pixels after normalization with the scales. The error is the distance between the center of detection to the groundtruth. Experimental results show that in certain frames, the tracking error is bigger. This is because the proposed algorithm tries to center at the pupil, instead of the center of the eyes.

5. DISCUSSION AND CONCLUDING REMARKS

A simultaneous eye tracking and blink detection system is presented in this paper. We used two interactive particle filters for this purpose, each particle filter serves to track the eye localization by exploiting AR models for describing the state transition and a classification-based model in tensor subspace for measuring the observation. One particle filter tracks the closed eyes and the other one tracks the open eyes. The set of particles that gives higher confidence is used to determine the estimated eye location as well as the eye's status (open versus closed); also the other set of particles is reinitialized accordingly. The system dynamics are described by two types of hidden state variables: the position and the scale. We use a second-order autoregression model for describing the eye's movement and a first-order autoregression model for describing the scale transition. Tensor subspace analysis is used for feature extraction and logistic regression is used to evaluate the posterior probabilities. The algorithm is evaluated using videos collected under different scenarios, including both indoor and outdoor data. We evaluated the performance from both the blink detection rate and the tracking accuracy perspective. Experimental setup for acquiring benchmark data to evaluate the accuracy is presented; and the experimental results are shown, which show that the proposed algorithm is able to accurately track eye locations and detect both voluntary long blinks and involuntary short blinks.

ACKNOWLEDGMENTS

This research was supported in part by grants from the UC Discovery Program and the Technical Support Working Group of the US Department of Defense. The authors are thankful for the assistance and support of their colleagues from the UCSD Computer Vision and Robotics Research Laboratory, especially valuable assistance provided by Shinko Cheng, which made systematic experimental evaluation using the motion capture system possible.

REFERENCES

[1] N. Kojima, K. Kozuka, T. Nakano, and S. Yamamoto, "Detection of consciousness degradation and concentration of a driver for friendly information service," in *Proceedings of the IEEE International Vehicle Electronics Conference*, pp. 31–36, Tottori, Japan, September 2001.

[2] P. Smith, M. Shah, and N. D. V. Lobo, "Monitoring head/eye motion for driver alertness with one camera," in *Proceedings of the International Conference on Pattern Recognition*, vol. 15, pp. 636–642, Cambridge, UK, September 2000.

[3] K. Grauman, M. Betke, J. Lombardi, J. Gips, and G. Bradski, "Communication via eye blinks and eyebrow raises: video-based human-computer interfaces," *Universal Access in the Information Society*, vol. 2, no. 4, pp. 359–373, 2003.

[4] K. Grauman, M. Betke, J. Gips, and G. R. Bradski, "Communication via eye blinks—detection and duration analysis in real time," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 1010–1017, Kauai, Hawaii, USA, December 2001.

[5] M. Chau and M. Betke, "Real time eye tracking and blink detection with usb cameras," Tech. Rep. 2005-12, Boston University Computer Science, Boston, Mass, USA, April 2005.

[6] T. Moriyama, T. Kanade, J. F. Cohn, et al., "Automatic recognition of eye blinking in spontaneously occurring behavior," in *Proceedings of the International Conference on Pattern Recognition (ICPR '02)*, vol. 16, pp. 78–81, Kauai, Hawaii, USA, 2002.

[7] D. Gorodnichy, "Second order change detection, and its application to blink-controlled perceptual interfaces," in *Proceedings of the International Association of Science and Technology for Development (IASTED '03) Conference on Visualization, Imaging and Image Processing (VIIP '03)*, pp. 140–145, Benalmadena, Spain, September 2001.

[8] T. Morris, P. Blenkhorn, and F. Zaidi, "Blink detection for real-time eye tracking," *Journal of Network and Computer Applications*, vol. 25, no. 2, pp. 129–143, 2002.

[9] J. F. Cohn, J. Xiao, T. Moriyama, Z. Ambadar, and T. Kanade, "Automatic recognition of eye blinking in spontaneously occurring behavior," 2007, to appear in *Behavior Research Methods, Instruments, and Computers*.

[10] J. C. McCall and M. M. Trivedi, "Facial action coding using multiple visual cues and a hierarchy of particle filters," in *Proceedings of the IEEE Workshop on Vision for Human Computer Interaction in Conjunction with IEEE (CVPR '06)*, vol. 2006, p. 150, New York, NY, USA, 2006.

[11] J. Wu and M. M. Trivedi, "Robust facial landmark detection for intelligent vehicle system," in *Proceedings of the IEEE International Workshop on Analysis and Modeling of Faces and Gestures in Conjunction with IEEE (ICCV '05)*, vol. 3723, pp. 213–228, Beijing, China, 2005.

[12] J. Wu and M. M. Trivedi, "A binary tree for probability learning in eye detection," in *Proceedings of the IEEE International Workshop on Face Recognition Grand Challenge in conjunction with IEEE (CVPR '05)*, vol. 3, p. 170, San Diego, Calif, USA, 2005.

[13] G. Welch and G. Bishop, "An introduction to the kalman filter," Tech. Rep., University of North Carolina at Chapel Hill, Chapel Hill, NC, USA, 1995.

[14] F. Gustafsson, F. Gunnarsson, N. Bergman, et al., "Particle filters for positioning, navigation, and tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 425–437, 2002.

[15] Y. Rui and Y. Chen, "Better proposal distributions: object tracking using unscented particle filter," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, vol. 2, pp. 786–793, 2001.

[16] M. Lee, I. Cohen, and S. Jung, "Particle filter with analytical inference for human body tracking," in *Proceedings of the IEEE Workshop on Motion and Video Computing*, pp. 159–165, December 2002.

[17] M. Bolić, S. Hong, and P. M. Djurić, "Performance and complexity analysis of adaptive particle filtering for tracking applications," in *Conference Record of the Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 853–857, 2002.

[18] C. Chang and R. Ansari, "Kernel particle filter: iterative sampling for efficient visual tracking," in *IEEE International Conference on Image Processing*, vol. 3, pp. 977–980, 2003.

[19] C. Chang and R. Ansari, "Kernel particle filter for visual tracking," *IEEE Signal Processing Letters*, vol. 12, no. 3, pp. 242–245, 2005.

[20] A. Giremus, A. Doucet, V. Calmettes, and J.-Y. Tourneret, "A rao-blackwellized particle filter for INS/GPS integration," in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '04)*, vol. 3, pp. 964–967, 2004.

[21] J. S. Liu and R. Chen, "Blind deconvolution via sequential imputation," *Journal of the American Statistical Association*, vol. 90, no. 430, pp. 567–576, 1995.

[22] A. Doucet, de Freitas, J. F. G., and N. J. Gordon, *Sequential Monte Carlo Methods in Practice*, Springer, New York, NY, USA, 2001.

[23] K. Heine, "Unified framework for sampling/importance resampling algorithms," in *Proceedings of the IEEE International Conference on Information Fusion*, vol. 2, pp. 1459–1464, 2005.

[24] J. S. Liu and R. Chen, "Sequential monte carlo methods for dynamic systems," *Journal of the American Statistical Association*, vol. 93, no. 443, pp. 1032–1044, 1998.

[25] R. Karlsson, *Particle Filtering for Positioning and Tracking Applications*, Ph.D. thesis, Linköping University, Linköping, Sweden, 2005.

[26] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *IEE Proceedings, Part F: Radar and Signal Processing*, vol. 140, no. 2, pp. 107–113, April 1993.

[27] M. K. Pitt and N. Shephard, "Filtering via simulation: auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.

[28] Z. Khan, T. Batch, and F. Dellaert, "A rao-blackwellized particle filter for eigen tracking," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 980–986, 2004.

[29] S. Sarkka, A. Vehtari, and J. Lampinen, "Rao-blackwellized particle filter for multiple target tracking," *Information Fusion*, vol. 8, no. 7, pp. 2–15, 2007.

[30] J. Carpenter, P. Clifford, and P. Fernhead, "An improved particle filter for non-linear problems," Tech. Rep., Department of Statistics, University of Oxford, Oxford, UK, 1997.

[31] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[32] J. Hol, T. Schon, and F. Gustafsson, "On resampling algorithms for particle filters," in *Nonlinear Statistical Signal Processing Workshop*, Cambridge, UK, September 2006.

[33] M. Isard and A. Blake, "Visual tracking by stochastic propagation of conditional density," in *Proceedings of the 4th European Conference on Computer Vision (ECCV '06)*, pp. 343–356, Graz, Austria, April 1996.

[34] M. Isard and A. Blake, "Condensation—conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[35] K. Nishiyama, "Fast and effective generation of the proposal distribution for particle filters," *Signal Processing*, vol. 85, no. 12, pp. 2412–2417, 2005.

[36] Y. Guan, R. Fleißner, P. Joyce, and S. M. Krone, "Markov chain Monte Carlo in small worlds," *Statistics and Computing*, vol. 16, no. 2, pp. 193–202, 2006.

[37] C. Shen, M. J. Brooks, and A. D. Van Hengel, "Augmented particle filtering for efficient visual tracking," in *Proceedings of the International Conference on Image Processing (ICIP '05)*, vol. 3, pp. 856–859, 2005.

[38] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe, "A boosted particle filter: multitarget detection and tracking," in *Proceedings of the European Conference on Computer Vision*, vol. 3021, pp. 28–39, Copenhagen, Denmark, May 2004.

[39] X. Xu and B. Li, "Head tracking using particle filter with intensity gradient and color histogram," in *Proceedings of the IEEE International Conference on Multimedia and Expo, (ICME '05)*, vol. 2005, pp. 888–891, 2005.

[40] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *Proceedings of the European Conference on Computer Vision (ECCV '02)*, Copenhagen, Denmark, May 2002.

[41] C. Yang, R. Duraiswami, and L. Davis, "Fast multiple object tracking via a hierarchical particle filter," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV '05)*, vol. 1, pp. 212–219, 2005.

[42] M. Pupilli and A. Calway, "Real-time camera tracking using a particle filter," in *Proceedings of the British Machine Vision Conference*, pp. 519–528, Oxford Brookes University, Oxford, UK, September 2005.

[43] M. Alex, O. Vasilescu, and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," in *Proceedings of the European Conference on Computer Vision (ECCV '02)*, pp. 447–460, Copenhagen, Denmark, May 2002.

[44] D. Cai, X. He, and J. Han, "Subspace learning based on tensor analysis," Tech. Rep. (UIUCDCS-R-2005-2572), Department of Computer Science,University of Illinois at Urbana-Champaign, Champaign, Ill, USA, 2005.

[45] X. He, D. Cai, and P. Niyogi, "Tensor subspace analysis," in *Proceedings of the Neural Information Processing Systems*, 2005.

[46] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[47] X. He, S. Yan, Y. Hu, P. Niyogi, and H.-J. Zhang, "Face recognition using laplacianfaces," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 3, pp. 328–340, 2005.

[48] S. Esaki, Y. Ebisawa, A. Sugioka, and M. Konishi, "Quick menu selection using eye blink for eye-slaved nonverbal communicator with video-based eye-gaze detection," in *Annual International Conference of the IEEE Engineering in Medicine and Biology*, vol. 5, pp. 2322–2325, 1997.

[49] D. Goryn and S. Hein, "On the estimation of rigid body rotation from noisy data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 12, pp. 1219–1220, 1995.