J. Vis. Commun. Image R. 20 (2009) 9–27

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Face detection and tracking using a Boosted Adaptive Particle Filter

Wenlong Zheng [a,*], Suchendra M. Bhandarkar [b]

[a] Department of Information Systems, Northwest Missouri State University, 800 University Drive, Maryville, MO 64468, USA
[b] Department of Computer Science, The University of Georgia, Athens, GA 30602-7404, USA

ARTICLE INFO

Article history:
Received 30 March 2007
Accepted 9 September 2008
Available online 17 September 2008

Keywords:
Adaptive Particle Filter
Particle filter
Face detection
Video tracking
Image analysis
Boosted learning
Boosted Adaptive Particle Filter
Adaptive Learning Constraint

ABSTRACT

A novel algorithm, termed a Boosted Adaptive Particle Filter (BAPF), for integrated face detection and face tracking is proposed. The proposed algorithm is based on the synthesis of an adaptive particle filtering algorithm and the AdaBoost face detection algorithm. An Adaptive Particle Filter (APF), based on a new sampling technique, is proposed. The APF is shown to yield more accurate estimates of the proposal distribution and the posterior distribution than the standard Particle Filter thus enabling more accurate tracking in video sequences. In the proposed BAPF algorithm, the AdaBoost algorithm is used to detect faces in input image frames, whereas the APF algorithm is designed to track faces in video sequences. The proposed BAPF algorithm is employed for face detection, face verification, and face tracking in video sequences. Experimental results show that the proposed BAPF algorithm provides a means for robust face detection and accurate face tracking under various tracking scenarios.

1. Introduction

Face detection is important in several automated systems that take as input images of the human face. Examples include fully automatic face recognition systems, video-based surveillance and warning systems, human face/body tracking systems and perceptual human–computer interfaces. Most face detection algorithms can be classified as feature-based or appearance-based. In recent years, appearance-based face detection algorithms that employ machine learning and statistical estimation methods have demonstrated excellent results among all existing face detection methods.

Examples of appearance-based face detection techniques include the AdaBoost algorithm [1,2], the FloatBoost algorithm [3], the S-AdaBoost algorithm [4], neural networks [5,6], Support Vector Machines (SVM) [7,8], Hidden Markov Models [9], and the Bayes classifier [10,11]. Viola and Jones [1,2] propose a robust AdaBoost face detection algorithm, which can detect faces in a rapid and robust manner with a high detection rate. Li et al. [3] propose the FloatBoost algorithm, an improved version of the AdaBoost algorithm, for learning a boosted classifier with minimum error rate. The FloatBoost algorithm uses a backtracking mechanism to improve the face detection rate after each iteration of the AdaBoost procedure. However this method is computationally more intensive than the AdaBoost algorithm. Jiang and Loe [4] propose the S-AdaBoost algorithm, a variant of the AdaBoost algorithm, for handling outliers in pattern detection and classification. Since the S-AdaBoost algorithm uses different classifiers in different phases of computation, it suffers from computational inefficiency and lack of accuracy.

Among the various neural network-based approaches to face detection, the work of Rowley et al. [5] is particularly well known. Rowley et al. [5] employ a multilayer neural network to learn the face and non-face patterns from training sets consisting of face and non-face images. A significant drawback of their technique is that the detection is limited to primarily upright frontal faces. Although Rowley et al. [5] further generalize their method to detect rotated face images, the reported results are not promising because of the resulting low detection rate. Face detection techniques based on Support Vector Machines (SVMs) use structural risk minimization to minimize the upper bound of the expected generalization error [7,8]. The major disadvantages of SVMs include intensive computation during the learning process and high memory requirement. Face detection techniques based on Hidden Markov Models (HMMs) assume that face and non-face patterns can be characterized as outputs of a parametric random Markovian process whose parameters can be learned using a well-defined estimation procedure [9]. The goal of training an HMM is to estimate the appropriate parameters in the HMM model that maximize the probability or likelihood of the observed training data. Schneiderman and Kanade [10] present a naive Bayes classifier for face detection, which is based on the estimation of the joint probability of the local appearance and position of a face pattern at multiple scales. However, the performance of the naive Bayes

* Corresponding author.
E-mail addresses: zheng@nwmissouri.edu (W. Zheng), suchi@cs.uga.edu (S.M. Bhandarkar).

1047-3203/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jvcir.2008.09.001

classifier is reported to be poor [10]. To address this problem, Schneiderman [11] has proposed a restricted Bayesian network for face detection based on performing a search in the large space of possible network structures to determine the optimal structure of a Bayesian network-based classifier.

Object tracking in video has also been studied extensively by researchers in computer vision because of various computer vision applications such as autonomous robots [12], video surveillance [13], human eye tracking [14] and human face tracking [15] that use tracking algorithms extensively. Object tracking algorithms designed to operate under more general and less structured situations need to deal with complex issues of uncertainty and error arising from occlusion, and changes in illumination, viewpoint and object scale [16]. Consequently, many techniques have been developed to tackle the various aforementioned issues in visual object tracking and reported in the literature over the past decade.

The various techniques for visual object tracking can be classified as image (region)-based, contour-based or filtering-based. Image (region)-based tracking methods typically extract generic region-based features from the input images and then combine these features using high-level scene information [16]. Intille et al. [16] propose a blob-tracker for human tracking in real time wherein the background is subtracted to extract foreground regions. The foreground regions are then divided into blobs based on color. This approach exhibits good run-time performance, but suffers from a major disadvantage in terms of merging of blobs when the objects in the scene approach each other. Contour-based tracking methods assume that the tracked objects are bounded by contours with known properties [17–19]. The contour pixels are tracked from one image frame to the next using a predefined contour shape model. Dynamical or elastic contour models are used to handle changes in object shape due to deformation or change in scale. Filtering-based methods are based on prediction and updating of object features over time, i.e., over successive frames in the video sequence. Tracking of object shape and object location over time is tackled adequately by the traditional Kalman filter in cases where the tracking problem can be effectively modeled as a linear dynamic system [20]. The Extended Kalman Filter (EKF) is an extension of the traditional Kalman filter to a non-linear but unimodal process where the non-linear behavior can be approximated by local linearization [21]. However, it is widely accepted that the particle filter is superior to the traditional Kalman filter in terms of tracking performance [22], since the particle filter provides a robust object tracking framework without being restricted to a linear system model.

Particle filters, also known as sequential Monte Carlo filters, have been widely used in visual tracking to address limitations arising from non-linearity and non-normality of the motion model [23,24]. The basic idea of the particle filter is to approximate the posterior density using a recursive Bayesian filter based on a set of particles with certain assigned weights. The Condensation algorithm, a simple particle filter, proposed by Isard [25] is designed to solve tracking problems arising from non-linearity and non-normality of the motion model. During the sampling step, the Condensation algorithm uses a simple proposal distribution to draw a set of particles, which defines the conditional distribution on the particle state in the previous frame. The proposal distribution is then used to approximate the target *a posteriori* distribution. A commonly observed shortcoming of the Condensation algorithm is that the proposal distribution does not incorporate the information from the current frame thus resulting in longer run time needed for convergence to the desired *a posteriori* distribution.

Various approaches have been developed to improve the tracking performance of the conventional particle filter. Li et al. [23] propose a Kalman particle filter (KPF) and an unscented particle filter (UPF) to improve the particle sampling procedure in the context of visual contour tracking. This approach makes use of a Kalman filter or an unscented Kalman filter to incorporate the current observation in the estimation of the proposal distribution. The Kalman filter or the unscented Kalman filter is shown to steer the set of particles to regions of high likelihood in the search space, and thus reducing the number of particles needed to estimate the proposal distribution [23]. To address the occlusion problem, Wang and Cheong [26] propose a particle filter with a Markov random field (MRF)-based representation of the tracked object within a dynamic Bayesian framework. This method transforms an object into a composite of multiple MRF regions to improve the modeling and tracking accuracy. Chang et al. [22] present a kernel particle filter to improve the sampling efficiency for multiple object tracking using data association techniques. This scheme invokes kernels to approximate continuously the posterior density, where the kernels for object representation and localization are allocated based on the value of the gradient derived from the kernel density. However, this method can not handle situations in which the motion pattern of objects in one group changes drastically. Rathi et al. [19] formulate geometric active contours as a parameterization technique to deal with deformable objects. However, their technique performs poorly when the tracked object is completely occluded over several frames. Isard and MacCormick [27] propose a Bayesian multiple blob-tracker (BraMBLe), an early variant of a particle filter, in which the number of tracked objects is allowed to vary during the tracking process. Nonetheless, this approach relies on modeling a fixed background to identify foreground objects, a situation that is not always practical in real-world tracking scenarios. To address this problem, Okuma et al. [24] relax the assumption of a fixed background and allow the background to vary in order to handle real-world image sequences. Okuma et al. [24] also propose a boosted particle filter (BPF) for multiple object detection and tracking, which interleaves the AdaBoost algorithm with a simple particle filter based on the Condensation algorithm. However, this method does not present a systematic way to incorporate object models to guarantee accurate approximation of the proposal distribution, nor does it address the occlusion problem.

In this paper, we propose a novel particle filtering scheme, termed as an Adaptive Particle Filter (APF), to enable estimation of the proposal distribution and the posterior distribution with a much higher degree of accuracy. Based on the previous work of Isard [25], Li et al. [23], Vermaak et al. [28] and Okuma et al. [24], we also propose a novel scheme for integration of face detection and face tracking by combining the APF-based tracking algorithm with the AdaBoost face detection algorithm. We term the combination of the proposed APF algorithm and the AdaBoost algorithm as a Boosted Adaptive Particle Filter (BAPF). In the proposed BAPF-based face tracking scheme, the AdaBoost algorithm is used to detect faces and also verify the existence of a tracked face in an input image frame, whereas the APF algorithm is designed to track the detected faces across the image frames in the video sequence. The BAPF algorithm is shown to yield very good tracking results in situations where the tracked objects are severely occluded. Experimental results show that the proposed BAPF scheme provides robust face detection and accurate face tracking in various tracking scenarios.

2. Statistical model

A glossary of mathematical notation used in the formulation of the statistical model and the particle filtering algorithm is given in Appendix A.

2.1. Observation model

We denote a state vector for an object by **x**, and an observation vector by **y**. It is important for contour tracking to obtain an accurate

estimate of the observation likelihood (also termed as the observation density) $p(\mathbf{y}|\mathbf{x})$. Blake et al. [29], Isard [25], and MacCormick and Blake [18,30] introduce statistical models for estimation of the observation density $p(\mathbf{y}|\mathbf{x})$. These models use specific image features that are collected along a set of normals to a hypothesized contour. A finite number of sample points, called control points, are generated on the hypothesized contour. We follow the general direction provided by the aforementioned models for modeling the observation process, but specifically follow the model proposed by MacCormick [18].

Fig. 1 shows an observed contour and image features extracted along a measurement line. We denote the finite number of sample points on a hypothesized contour by a set $\{x_i;\ i = 1,2,\ldots,n\}$. We term the normals to the above contour as measurement lines, and denote them by a set $\{s_i;\ i = 1,2,\ldots,n\}$. The length of the measurement lines is fixed at a value T. The Canny edge detector is applied to the measurement line s_i $(i = 1,2,\ldots,n)$ in order to obtain the positions of the edge features $\{z_i^{(m)},\ m = 1,2,\ldots,m_i\}$ where m is the index of the detected features, and m_i is the number of detected features. As can be seen, each feature is jointly generated by the boundary of an object and random clutter present in the image.

The clutter features on the measurement line s_i $(i = 1,2,\ldots,n)$ are assumed to obey a Poisson distribution with density parameter λ given by

$$p_T(m_i) = \frac{(\lambda T)^{m_i}}{m_i!}\mathrm{e}^{-\lambda T} \tag{1}$$

where m_i is the number of detected clutter features. A boundary density function is assumed to obey a Gaussian distribution; thus the generic likelihood function of an observation at a sample point x_i $(i = 1,2,\ldots,n)$ can be described by [18]:

$$p_{x_i}(\mathbf{z}|\mathbf{v} = \{x_i\}) = \frac{(\lambda T)^{m_i}}{m_i!}\mathrm{e}^{-\lambda T}\left(q_0 + \frac{q_1}{\lambda}\sum_{i=1}^{m_i}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(z_i^{(m)} - x_i)^2}{2\sigma^2}\right)\right) \tag{2}$$

where q_0 is the probability of undetected features for an object boundary, and q_1 is the probability of detected features for an object boundary. Assuming that the sample points are independent and identically distributed, the overall likelihood function of the observation $p(\mathbf{y}|\mathbf{x})$ can be represented as

$$p(\mathbf{y}|\mathbf{x}) = \prod_{i=1}^{n}\frac{(\lambda T)^{m_i}}{m_i!}\mathrm{e}^{-\lambda T}\cdot\prod_{\text{meas. line } s_i \text{ intersecting } x}\left(q_0 + \frac{q_1}{\lambda}\sum_{i=1}^{m_i}\frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{(z_i^{(m)} - x_i)^2}{2\sigma^2}\right)\right) \tag{3}$$

2.2. Dynamical model

Generally speaking, a particle filter algorithm requires a dynamical model to describe the evolution of a tracking system over time. An auto-regressive process (ARP) model has been widely used for the purpose of formulating such a dynamical model [18,31]. Blake et al. [29,32] model the object dynamics as a second-order process. Isard and Blake [33] and Li et al. [23] also follow the dynamical model of Blake et al. [29,32] in their work on object tracking. Following the previous works of Blake et al. [29,32], Isard and Blake [33] and Li et al. [23], this paper also employs a second-order ARP as the dynamical model for face tracking. It is widely accepted that the second-order ARP captures the various motions of interest that are relevant to visual tracking [18]. The parameters of the dynamical model in a typical real-world application can be obtained via learning from the input training data. The second-order ARP represents the state $\mathbf{x}_t$ at time t as a linear combination of the previous two states and additive Gaussian noise. The dynamical model can be represented as a second-order linear difference equation:

$$\mathbf{x}_t - \bar{\mathbf{x}} = \mathbf{A}(\mathbf{x}_{t-1} - \bar{\mathbf{x}}) + \mathbf{B}\boldsymbol{\omega}_t \tag{4}$$

where $\boldsymbol{\omega}_t$ is Gaussian noise that is independent of the state vector $\mathbf{x}_t$, $\bar{\mathbf{x}}$ denotes the mean value of the state vector, and $\mathbf{A}$ and $\mathbf{B}$ are matrices describing the deterministic component and the stochastic component of the dynamical model, respectively. The state vector $\mathbf{x}_t$ encodes the knowledge of the object contour in the current state and the previous state and is represented by

$$\mathbf{x}_t = \begin{pmatrix}\mathbf{X}_{t-1}\\ \mathbf{X}_t\end{pmatrix}.$$

In most real-world applications, one can set some reasonable default values for the parameters $\mathbf{A}$, $\mathbf{B}$ and $\bar{\mathbf{x}}$ of the dynamical model. However, it is more effective to approximate these values via observation of video sequences, in which the tracked object undergoes some typical representative motions [29,34]. The dynamical model can also be represented by a temporal Markov chain [33] given by

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = C\cdot\exp\left(-\frac{1}{2}|\mathbf{B}^{-1}((\mathbf{x}_t - \bar{\mathbf{x}}) - \mathbf{A}(\mathbf{x}_{t-1} - \bar{\mathbf{x}}))|^2\right) \tag{5}$$

where C is a constant, and $|\cdot|$ denotes the Euclidean norm.

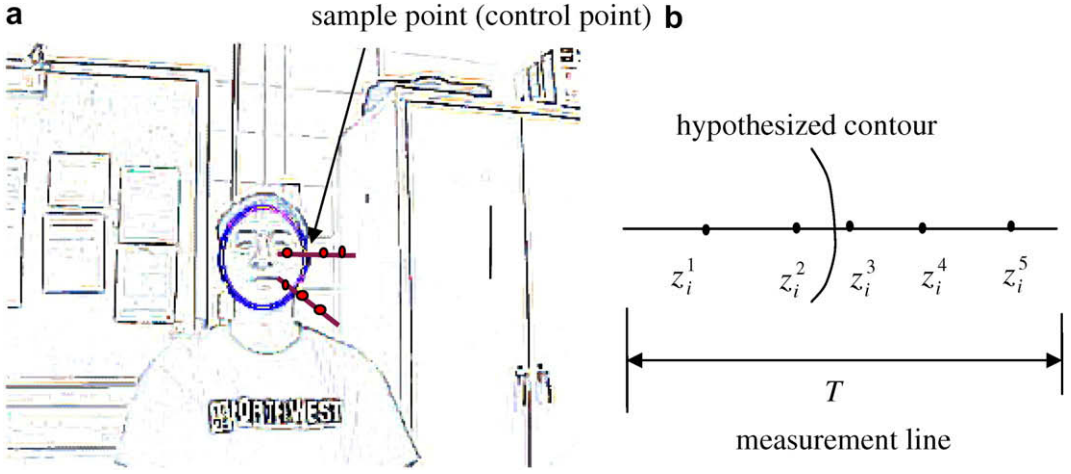

Fig. 1. (a) Observation process: the ellipse is a hypothesized contour in an image. (b) The image features on the measurement line.

3. Face tracking using particle filtering

3.1. The filtering distribution

We denote a state vector for an object at time t by $\mathbf{x}_t$, and its history up to time t by $\mathbf{x}_{1:t} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t\}$. Likewise, an observation vector at time t is denoted by $\mathbf{y}_t$ and its history up to time t is denoted by $\mathbf{y}_{1:t} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t\}$. The standard problem of target tracking, in the terminology of statistical pattern recognition, is to estimate the state $\mathbf{x}_t$ of the objects at time t, using a set of observations y_t from a sequence of input images. The *a posterior* density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ represents all the information about $\mathbf{x}_t$ at time t that is potentially deducible from the set of observations $\mathbf{y}_t$ up to that time.

We assume that the object dynamics constitute a temporal Markov process and that the observations $\mathbf{y}_t$ are independent. Therefore, the dynamics are determined by a transition prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. Given the transition prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and the observation density $p(\mathbf{y}_t|\mathbf{x}_t)$, the posterior density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ can be computed by applying Bayes' rule [35] for inferring the posterior state density from time-varying observations. The posterior density is estimated recursively via Bayesian filtering [33,36] as follows

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t, \mathbf{y}_{1:t-1})p(\mathbf{x}_t|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})} = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})} \tag{6}$$

where

$$p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1} \tag{7}$$

The posterior density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is generally evaluated in two steps, namely the prediction step and the updating step. First, an effective prior $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ shown in Eq. (7) is predicted from the posterior density $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ via the transition prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. Second, the posterior density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is updated based upon the new observation $\mathbf{y}_t$ at time t, as given by Eq. (6).

The observation prior $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$, which is the denominator in Eq. (6), can be represented by

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) = \sum_{\mathbf{x}_t} p(\mathbf{y}_t, \mathbf{x}_t|\mathbf{y}_{1:t-1}) = \sum_{\mathbf{x}_t} p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) \tag{8}$$

Furthermore, the observation prior $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$ can be represented by the following integral:

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_t \tag{9}$$

Thus Eq. (6) becomes

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})}{\int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_t} \tag{10}$$

Using Eq. (7), we substitute the expression for the effective prior $p(x_t|y_{1:t-1})$ in Eq. (10) to obtain

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{t-1}}{\int p(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{t-1}\mathrm{d}\mathbf{x}_t} \tag{11}$$

In addition to the estimate of the posterior density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$, an estimate of a function $f(\mathbf{x}_t)$ of object state vector is also required under many situations. The expected value or estimate of the function $f(\mathbf{x}_t)$ is approximated as

$$E[f(\mathbf{x}_t)] = \int f(\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{y}_{1:t})\mathrm{d}\mathbf{x}_t \tag{12}$$

Eqs. (10) and (11) represent an optimal solution to the standard problem of object tracking via recursive Bayesian filtering. Obviously, this solution involves the computation of high-dimensional integrals, and dealing with the non-linearity and non-normality of the motion model in many tracking scenarios. High-dimensional integrations usually cannot be easily computed in a closed analytical form. Thus a particle filter, also known as a sequential Monte Carlo filter, is adopted as a practical solution to the otherwise intractable problem of object tracking via recursive Bayesian filtering.

3.2. The standard particle filter

A standard particle filter uses Monte Carlo simulation to estimate the posterior probability density function $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ given in Eq. (10). Particle filtering uses random sampling strategies to model a complex posterior probability density function $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. It uses N weighted discrete particles to approximate the posterior probability density function $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ via observation of the data. Each particle consists of a state vector $\mathbf{x}$ and a weight w. The weighted particle set is given by $\{(\mathbf{x}_t^{(i)}, w_t^{(i)}),\ i = 1, 2, \ldots, N\}$. Particle filtering samples the space spanned by $\mathbf{x}_t$ with N discrete particles and approximates the distribution using the discrete points sampled by the particles and their associated weights. Specifically, we assume that N particles are used in the sampling procedure to approximate the posterior probability density function $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$, and that the N discrete sample points in the space of $\mathbf{x}_{t-1}$ are given by $\mathbf{x}_{t-1}^1, \mathbf{x}_{t-1}^2, \ldots, \mathbf{x}_{t-1}^N$, respectively. Thus we have

$$p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} w_{t-1}^i \delta(\mathbf{x}_{t-1} - \mathbf{x}_{t-1}^i) \tag{13}$$

Since it is not feasible to draw samples directly from the posterior distribution, a proposal distribution $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_{1:t})$ is used to draw the samples easily for approximation of the posterior probabilities. Using the proposal distribution $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_{1:t})$, a particle filter generates the ith particle $\mathbf{x}_t^{(i)}$ from $\mathbf{x}_{t-1}^{(i)}$, where $(i = 1, 2, \ldots, N)$, and computes the weight for $\mathbf{x}_t^{(i)}$ using the following equation

$$w_t^{(i)} = \frac{p(\mathbf{y}_t|\mathbf{x}_t^{(i)})p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)})}{q(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_{1:t})} w_{t-1}^{(i)} \tag{14}$$

The posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ can be thus approximated as

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)}) \tag{15}$$

The estimate of the function $f(\mathbf{x}_t)$ of the state vector could be then computed as

$$E[f(\mathbf{x}_t)] \approx \sum_{i=1}^{N} w_t^{(i)} f(\mathbf{x}_t^{(i)}). \tag{16}$$

3.3. The Adaptive Particle Filter

One of the active research areas in particle filtering in recent years is the generation of a good proposal distribution $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_{1:t})$ that would enable a more accurate estimate of the posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. The aim is to obtain as close an approximation to the posterior probability distribution as possible. The standard particle filter based on the Condensation algorithm [33] does not use the knowledge obtained from the current image frame, which leads to higher inaccuracy in the estimate of the posterior distribution. Doucet et al. [36] propose an optimal proposal distribution (OPD) for state estimation of jump Markov linear systems, and recursively compute optimal state estimates based on the selection of the minimum variance of weights $w_t^{(i)}$

$(i = 1, 2, \ldots, N)$. To overcome the problem of inefficient computation of the OPD, Li et al. [23] propose a Kalman particle filter (KPF) and an unscented particle filter (UPF) to drive a set of particles to the high-likelihood regions in the search space. Li et al. [23] propose a local linearization of the OPD to estimate the proposal distribution, which is assumed to be a Gaussian distribution. Therefore, the proposal distribution obtained by Li et al. [23] can be represented as

$$u_l(\mathbf{x}) = q(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1}^{(i)}, \mathbf{y}_{1:k}) = \mathcal{N}(\hat{\mathbf{x}}_t^{(i)}, \hat{\mathbf{P}}_t^{(i)}) \quad i = 1, 2, \ldots, N. \tag{17}$$

where $\hat{\mathbf{x}}_t^{(i)}$ and $\hat{\mathbf{P}}_t^{(i)}$ denote the mean and covariance, respectively, of the Gaussian distribution $\mathcal{N}(\hat{\mathbf{x}}_t^{(i)}, \hat{\mathbf{P}}_t^{(i)})$.

In this paper, we propose a new particle filtering scheme, termed as an Adaptive Particle Filter (APF), which enables significantly more accurate estimation of the proposal distribution and the posterior distribution. The proposed APF can be viewed as an extension of the Condensation algorithm and the Kalman particle filter to obtain an accurate approximation of the proposal distribution and the posterior distribution. The proposed APF algorithm is depicted in Fig. 2.

In the sampling step of the APF algorithm (Fig. 2), a new sampling strategy is used to improve the accuracy of the approximation, which is distinct from the sampling strategies used in other particle filters. The sampling step is considered to be the most important step in a particle filtering algorithm. For each discrete particle $\mathbf{x}_{t,l-1}^{(i)}$, the Adaptive Particle Filter generates a new particle $\mathbf{x}_{t,l}^{(i)}$ based on a proposal distribution $u_l(\mathbf{x})$. We use the loop controlled by the parameter l in the APF algorithm (Fig. 2) to implement the proposed sampling strategy. Let the parameter L denote the fixed number of iterations of loop l, where the value of L can be tuned based on the needs of different real-world applications. When $L = 1$, the APF is tantamount to the standard particle filter; thus the standard particle filter is a special case of the proposed APF. When $L > 1$, the APF performs more sampling iterations than the standard particle filter. We prove in Appendix A that the additional iterations of the APF result in a lower estimation error for the proposal distribution and posterior distribution.

In order to enable more accurate estimation of the proposal distribution, we iterate the sampling procedure subject to a constraint termed as the Adaptive Learning Constraint (ALC) described using Eq. (17). A more detailed explanation and derivation of the ALC is provided in Appendix A.

$$K_l \cdot \max_l \leq \alpha \cdot K_{l-1} \cdot \min_{l-1} \tag{18}$$

where

$$\max_l = \max_{1 \leq i \leq N} \{M_1|\xi_1^{(i)} - \mathbf{x}_{t,l}^{(i)}|, M_2|\xi_2^{(i)} - \mathbf{x}_{t,l}^{(i)}|\}$$

$$\min_{l-1} = \min_{1 \leq i \leq N} \{m_1|\xi_1^{(i)} - \mathbf{x}_{t,l-1}^{(i)}|, m_2|\xi_2^{(i)} - \mathbf{x}_{t,l-1}^{(i)}|\}$$

K_l, K_{l-1} are constants, and $0 < \alpha < 1$.

As long as the proposed ALC is satisfied, the current iteration that generates new particles for tracking in the current image frame will continue. The current iteration in the sampling step is halted when the proposed ALC ceases to be satisfied or the predefined loop threshold is reached. From both, a theoretical and practical viewpoint, the particles (and their associated weights and state vector) resulting from the latest iteration of the ALC represent a better approximation to the proposal distribution and the posterior distribution. The theoretical analysis and experimental results presented in the following sections of the paper confirm that the performance of the APF algorithm is indeed superior to that of the conventional particle filtering algorithm.

The other steps in the proposed APF algorithm are similar to those in the case of other particle filters such as the KPF and the UPF. The initialization step in the proposed APF algorithm takes advantage of the information derived from the results of the AdaBoost face detection algorithm.

4. The Boosted Adaptive Particle Filter

The proposed Boosted Adaptive Particle Filter (BAPF) for face detection and tracking employs two object models: the contour-based model used in the Adaptive Particle Filter (APF) and the region-based model used in the AdaBoost face detection procedure. The object models used in the context of tracking can be typically classified into three general categories [37]: contour-based models [23,38,39], region-based models [15,27,40], and feature point-basedmodels [41–43,45].

Since the proposed BAPF algorithm uses two distinct models for face detection and tracking, it has distinct advantages over the conventional particle filter. The incorporation of the AdaBoost algorithm within the APF algorithm is shown to substantially improve the robustness of the resulting BAPF algorithm. The AdaBoost algorithm provides a natural mechanism for integration of the contour- and region-based representations. This makes the proposed BAPF algorithm more powerful than the naïve K-means clustering method proposed by Vermaak et al. [28] in their distribution mixture tracking algorithm. The proposed BAPF algorithm also performs better than the distribution mixture representation scheme of Okuma et al. [24] since the proposed approach employs a more effective particle filtering algorithm, i.e., the APF algorithm. In the proposed BAPF scheme, the AdaBoost algorithm allows for explicit detection of faces entering and leaving the field of view of the camera, thus providing a means for (re)initialization of the APF-based tracking algorithm. The AdaBoost algorithm provides a means of verification of the predictions made by the APF-based tracking algorithm whereas the APF-based tracking algorithm, in turn, provides the focus-of-attention regions for the AdaBoost algorithm. The resulting BAPF algorithm is seen to provide robust face detection and accurate face tracking under various tracking scenarios.

4.1. Face detection using adaboost

Among the various face detection methods described in the published research literature, the boosted learning-based face detection methods have demonstrated excellent results. Building on the previous work of Tieu and Viola [44] and Schneiderman [11], Viola and Jones [1,2] have proposed a robust AdaBoost face detection algorithm, which can detect faces in an input image in a rapid and robust manner with a high detection rate. The face detection technique in AdaBoost is comprised of three aspects: the integral image, a strong classifier comprising of weak classifiers based on the AdaBoost learning algorithm, and an architecture comprising of a cascade of a number of strong classifiers.

We employ the AdaBoost scheme of Viola and Jones [1,2] for face detection. A 25-layer cascade of boosted classifiers is trained to detect multi-view faces. A set of sample face and non-face (termed as background) images are used for training. Another set of non-face sample images is collected from video sequences containing no faces. The results of the AdaBoost training procedure and the AdaBoost face detection procedure are presented in Section 5.

4.2. Integrating the Adaptive Particle Filter with adaboost

The proposed face detection and tracking scheme is based on the integration of two distinct models: an AdaBoost face detection model and an Adaptive Particle Filter (APF)-based face tracking

1. Initialization
   Initialize a set of particles from the prior $p(\mathbf{x}_0)$ to get $\{(\mathbf{x}_0^{(i)}, w_0^{(i)}), i = 1,2,\cdots,N\}$. Let *t=0*.
2. Sampling step
   1) For *l = 1, 2, … , L*
      a) For *i = 1, 2, … , N*
         Sample $\mathbf{x}_{t,l}^{(i)}$ from $\mathbf{x}_{t,l-1}^{(i)}$ based on the proposal distribution $u_l(\mathbf{x})$, where $u_l(\mathbf{x}) = q(\mathbf{x}_t^{(i)} \mid \mathbf{x}_{t-1}^{(i)}, \mathbf{y}_{1:k}) = \mathcal{N}(\hat{\mathbf{x}}_t^{(i)}, \hat{\mathbf{P}}_t^{(i)})$. Construct $p_{l-1}(\mathbf{x}) = \sum_{i=1}^{N} w_{t,l-1}^{(i)} \delta(\mathbf{x}_t - \mathbf{x}_{t,l-1}^{(i)})$, where $\delta(\cdot)$ is the Dirac function.
      b) If the Adaptive Learning Constraint is satisfied, where $K_l \cdot max_l \le \alpha \cdot K_{l-1} \cdot min_{l-1}$:
         i) Compute the weights of particles
            $w_{t,l}^{(i)} = p(\mathbf{y}_t \mid \mathbf{x}_{t,l}^{(i)}) p(\mathbf{x}_{t,l}^{(i)} \mid \mathbf{x}_{t-1}^{(i)}) w_{t,l-1}^{(i)}$, *i = 1, 2, … , N*
            $w_{t,0}^{(i)} = w_{t-1}^{(i)}$ when *l* = 1.
         ii) Normalize
            $w_{t,l}^{(i)} = \dfrac{w_{t,l}^{(i)}}{\sum_{i=1}^{N} w_{t,l}^{(i)}}$, *i = 1, 2, …, N*
         iii) Continue the loop *l*
      c) If the Adaptive Learning Constraint is not met, where $K_l \cdot max_l > \alpha \cdot K_{l-1} \cdot min_{l-1}$:
         i) Let $w_t^{(i)} = w_{t,l-1}^{(i)}$, $\mathbf{x}_t^{(i)} = \mathbf{x}_{t,l-1}^{(i)}$,
         ii) Break the loop *l*
   2) Let $w_t^{(i)} = w_{t,L}^{(i)}, \mathbf{x}_t^{(i)} = \mathbf{x}_{t,L}^{(i)}$, *i = 1, 2, … , N.*
   3) $w_t^{(i)} = \dfrac{w_t^{(i)}}{q(\mathbf{x}_t^{(i)} \mid \mathbf{x}_{t-1}^{(i)}, \mathbf{y}_{1:k})}$, *i = 1, 2, … , N.*
3. Estimation step
   Obtain a set of particles $\{(\mathbf{x}_t^{(i)}, w_t^{(i)}), i = 1,2,\cdots,N\}$. The posterior distribution $p(\mathbf{x}_t \mid \mathbf{y}_{1:t}) \approx \sum_{i=1}^{N} w_t^{(i)} \delta(\mathbf{x}_t - \mathbf{x}_t^{(i)})$ can be approximated using the output set of particles. The estimate value of $f(\mathbf{x}_t)$ can be computed as:
   $E[f(\mathbf{x}_t)] \approx \sum_{i=1}^{N} w_t^{(i)} f(\mathbf{x}_t^{(i)})$.
4. Selection step
   Resample particles $\mathbf{x}_t^{(i)}$ with probability $w_t^{(i)}$ to obtain *N* i.i.d random particles $\mathbf{x}_t^{(i)}$, approximately distributed with respect to $p(\mathbf{x}_t \mid \mathbf{y}_{1:t})$.
   Assign $w_t^{(i)} = \dfrac{1}{N}$, *i = 1, 2, … , N.*
5. Set *t=t+1*, go to step 2.

Fig. 2. The algorithm describing the Adaptive Particle Filter.

model. The AdaBoost face detection model performs multi-view face detection based on the trained AdaBoost algorithm. The APF model performs visual contour tracking using the adaptive particle filtering algorithm described in Section 3.3. Fig. 3 shows the closed-loop control system view of the proposed integrated face detection and face tracking scheme.

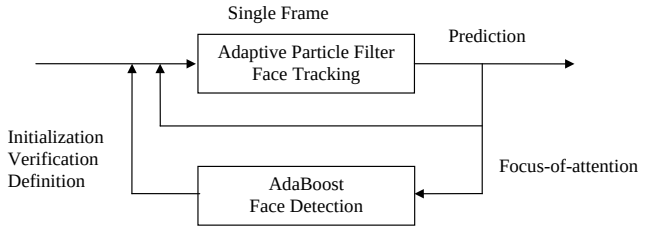

Fig. 3. Integration of the APF with AdaBoost within a single feedback control system.

The process for face detection and tracking contains two phases: an initialization phase and a tracking phase. In the initialization phase of the BAPF, the AdaBoost face detection model is used to provide the initial parameters for the APF face tracking model based on observations of the image frames in the video stream over a certain time interval. During the tracking phase, the AdaBoost face detection model and the APF face tracking model improve the tracking performance via synergetic interaction. The AdaBoost face detection model helps the APF face tracking model to find and define new objects (faces), and to verify the current states of the objects (faces) being tracked. On the other hand, the APF face tracking model provides focus-of-attention regions within the image to speed up the AdaBoost face detection procedure.

After performing the AdaBoost face detection procedure on an image, we obtain a confidence measure η for each detected face in the image. Likewise, using the APF tracking algorithm, the estimate of $f(\mathbf{x}_t)$ generated by the Adaptive Particle Filter at each sample point along the contour is given by

$$E[f(\mathbf{x}_t)] \approx \sum_{i=1}^{N} w_t^{(i)} f(\mathbf{x}_t^{(i)}) \tag{19}$$

The results of the AdaBoost algorithm and the APF algorithm are combined to update the position of a sampled point, as follows

$$E_c(f(\mathbf{x}_t)) = (1-\gamma)\cdot E(f(\mathbf{x}_t)) + \gamma \cdot \eta \cdot d \tag{20}$$

where $E_c(.)$ represents the estimated position of a sampled point on the contour combining the estimation values from the APF and the AdaBoost algorithm, the parameter γ is the weight assigned to the Adaboost detection procedure, the parameter η is a confidence measure assigned to each detected face in the image, d is the distance between the center of a detected face from the AdaBoost algorithm and the center of a sampled template contour from the APF algorithm, and $f(\mathbf{x}_t)$ represents the location of the contour of the tracked face. In practice, the output of the Adaboost face detection procedure is averaged over the previous F frames before combining it with the APF output using Eq. (19). The value of $E_c(f(\mathbf{x}_t))$ is fed back to the APF for further processing in successive iterations.

Eq. (19) demonstrates that the combination of the AdaBoost algorithm and the APF algorithm can be employed to obtain the estimated position of a sampled point on the contour. The actual estimated position was computed by adding a weighted estimate of the APF and a weighted distance between the center of a face detected using the AdaBoost algorithm and the center of a sampled template contour from the APF algorithm. The first term represents the influence of the APF, whereas the second term represents the influence of the AdaBoost algorithm. The parameter γ in Eq. (19) can be fine-tuned without affecting the convergence of the APF. When $\gamma = 0$, the BAPF degenerates to the pure APF. By increasing the value of γ, we lay a greater emphasis on the output of the AdaBoost face detection algorithm. On the other hand, when $\gamma = 1$, the BAPF degenerates to the pure AdaBoost algorithm, which does not take into account the output of the APF-based tracker. In practice, one could adjust the value of the parameter γ based on different scene conditions determined by the ambient illumination, clutter, and inter-object and background occlusions.

5. Experimental results

5.1. Adaboost face detection

The AdaBoost scheme of Viola and Jones [1,2] is used to detect faces in input images. In our experiment, we train a 25-layer cascade of strong classifiers to detect multi-view faces in video sequences. The training data set is composed of face and non-face images of size 20×20 pixels. A set of 6230 multi-view face images of eight persons is collected from video sequences under different conditions of ambient scene illumination, surface reflection, facial pose, facial expression and background composition in order to make face detection scheme more robust in different scenarios. The face images are cropped and scaled to a resolution of 20×20 pixels. Another set of 6598 non-face sample images of size 320×240 pixels are collected from video sequences containing no faces. The non-face sample images are of the same size as the video frames acquired by the video camera for real-time face tracking, but this is not a strict requirement. Fig. 4 shows some random face samples used for the training, and Fig. 5 shows some random non-face samples used for training. A larger training set of face and non-face examples typically leads to better detection results, although detection failures still exist in the input image regions that contain overlaps or occlusions, and are characterized by high scene clutter. Some results of face detection using our trained AdaBoost face detection procedure are illustrated in Fig. 6. AdaBoost face detection performs well in most cases, but generates false positives in very busy images that contain overlaps or occlusions, and are characterized by high scene clutter.

5.2. Boosted Adaptive Particle Filter

The proposed Boosted Adaptive Particle Filter (BAPF) is implemented using C++ under the Microsoft Visual C++ .NET environment on a 1.6 GHz Pentium-M workstation. Video sequences are of size 320×240 pixels and are sampled at 30 frames per second. In the beginning, the AdaBoost face detection algorithm initializes the object state(s) in the Adaptive Particle Filter (APF)-based face tracking algorithm. The initialization is done using observations of the image frames in the input video sequence over a certain time interval. Since the contour defining the appearance of the face in the video sequences is roughly circular or elliptical in shape, we use a simple parameterized model to represent the contour i.e., $Ax^2 + By^2 + C = 0$. Note that the proposed BAPF algorithm can also be used in the case of more complex contours that are modeled using a B-spline representation. The proposed BAPF algorithm has been applied to various tracking scenarios shown in Fig. 7 through Fig. 12. The tracking results are presented for three test video sequences that are captured under varying conditions defined by ambient scene illumination, surface reflection, object scale, facial pose (consisting of both, in-plane and out-of-plane rotations), facial expression, occlusions and background composition. In two of the test videos (test videos 1 and 3) there is a single face in the scene. Test video 1 is used in experiments comprising of different tracking scenarios, whereas test video 3 is used to compare the tracking accuracy of the proposed BAPF algorithm with that of the Condensation algorithm (i.e., the conventional particle filter). The third test video (test video 2), in which there are two faces in the scene, is used in experiments that cover different multi-face tracking scenarios. All tracking results are obtained using $N = 1000$ particles in the APF, BAPF and Condensation algorithms.

Fig. 4. Face examples.

Fig. 5. Non-face examples.

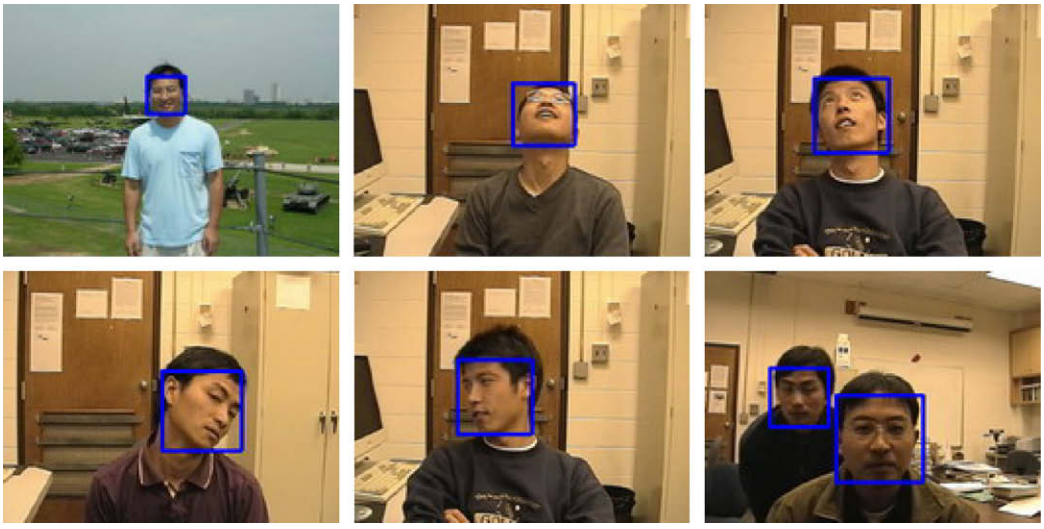

Fig. 6. Results of frontal face detection and multi-view face detection.

Fig. 7. Tracking results with scale changes on test video 1. From left to right, the frame numbers are 981, 1043 and 1067.

Fig. 8. Tracking results with illumination changes on test video 1. From left to right, the frame numbers are 866, 954 and 969.

5.2.1. Description of BAPF experiments

In Fig. 7 through 12, a yellow ellipse around a face implies the absence of occlusion, whereas a red ellipse indicates the presence of occlusion. Fig. 7 shows snapshots of a single-face tracking experiment on test video 1 where the scale of the tracked face is observed to change significantly. The tracking results show that the proposed BAPF tracking algorithm can handle significant scale changes in the object appearance. Fig. 8 shows snapshots of a single-face tracking experiment on test video 1 under changing illumination conditions. It demonstrates that the proposed BAPF algorithm is robust to various changes in ambient illumination conditions which can be attributed to the inherent robustness of the APF algorithm and the integration of the statistical learning procedure in AdaBoost with the APF-based tracker. Fig. 9 shows snapshots of a single-face tracking experiment on test video 1 under changes in viewpoint and facial pose arising from out-of-plane rotations of the face. It shows that the proposed BAPF algorithm can handle multi-view face detection and tracking. Fig. 10 shows snapshots of a single-face tracking experiment on test video 1 with in-plane rotations of the face. It shows that the proposed BAPF algorithm can handle object appearance changes due to in-plane object rotations. Fig. 11 shows snapshots of a single-face tracking experiment on test video 1 where the face is periodically occluded. It confirms that the proposed BAPF algorithm performs correctly in the presence of occlusions because of the robustness of the APF. Fig. 12 presents snapshots of a two-face tracking experiment on test video 2 which contains instances of inter-object (i.e., inter-face) occlusion. It can be observed that the proposed BAPF algorithm can deal with and recover from instances of inter-object (inter-face) occlusion in the video stream.

The tracking performance of the proposed BAPF algorithm is compared with that of the Condensation algorithm [33], which is a well-known implementation of a conventional particle filter-based tracker. Both algorithms employ $N = 1000$ particles for face tracking in test video 3. The experimental results show that the tracking accuracy of the proposed BAPF algorithm is superior to that of the Condensation algorithm. Thus, in the context of object tracking, the proposed BAPF algorithm can be deemed to yield better performance than the conventional particle filter. However, the better performance of the proposed BAPF algorithm comes at the price of computational efficiency. The BAPF algorithm actually consumes more computing resources than the relatively straightforward Condensation algorithm since the BAPF algorithm performs more computation per iteration in order to yield more accurate non-linear estimations of the object state(s). The tracking accuracy of the proposed BAPF algorithm and the Condensation algorithm can be visually compared using the snapshots shown in Figs. 13 and 14, respectively.

5.2.2. BAPF performance analysis

Using tracking accuracy and computation time as the performance metrics, we quantitatively analyze and compare the performance of the BAPF algorithm, APF algorithm, and Condensation algorithm. The tracking accuracy is defined in terms of the displacement error between the centroid of a ground truth face and the centroid of a tracked face in a video sequence. All three aforementioned tracking algorithms employ $N = 1000$ particles for face tracking and are tested on test video 3. In the following empirical comparison, the performance of the APF algorithm is compared to that of the Condensation algorithm, the performance of the BAPF algorithm is compared to that of the APF algorithm, the performance of the APF algorithm is analyzed for different values of the parameter L, the performance of the BAPF algorithm is analyzed for different values of the parameter F, and the performance of the BAPF algorithm is analyzed for different values of the parameter γ.

We first compare the performance of the APF algorithm to that of the Condensation algorithm. Both algorithms employ $N = 1000$ particles for face tracking in the test video 3. In the APF algorithm, the upper bound on the number of iterations of the loop controlled by parameter l is $L = 3$. The experimental results, as shown in Fig. 15 and Table 1, demonstrate that the tracking accuracy of the APF algorithm is superior to that of the Condensation algorithm. It can be seen from Table 1 that the mean displacement error in the case of the APF algorithm is significantly lower than that in the case of the Condensation algorithm. However, the tracking speed (in frames/s) of the APF algorithm is observed to be slower than that of the Condensation algorithm. As mentioned previously, the slower tracking speed of the APF algorithm can be attributed to the fact that the APF algorithm performs more computation per

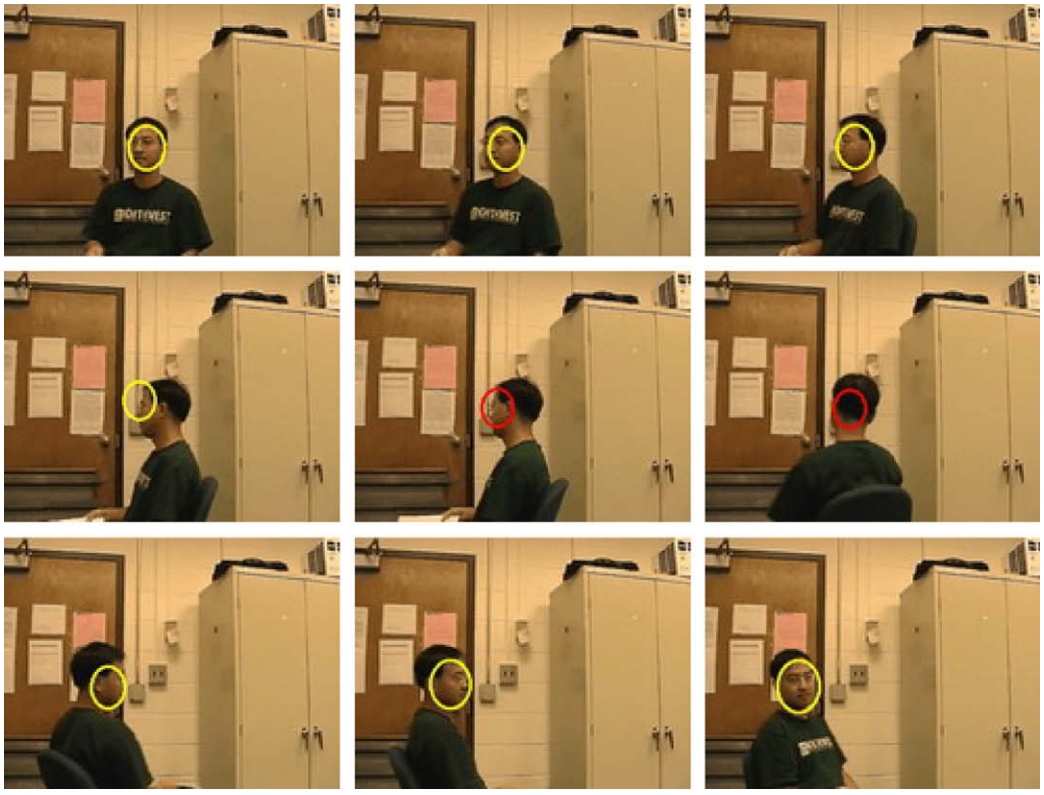

Fig. 9. Tracking results in the presence of multiple facial views and out-of-plane face rotations on test video 1. The yellow ellipse implies that no occlusion has occurred, whereas the red ellipse implies that occlusion has occurred. From top left to bottom right, the frame numbers are 515, 519, 524, 530, 533, 544, 566, 573 and 585.

Fig. 10. Tracking results in the presence of in-plane rotations on test video 1. From left to right, the frame numbers are 104, 135 and 152.

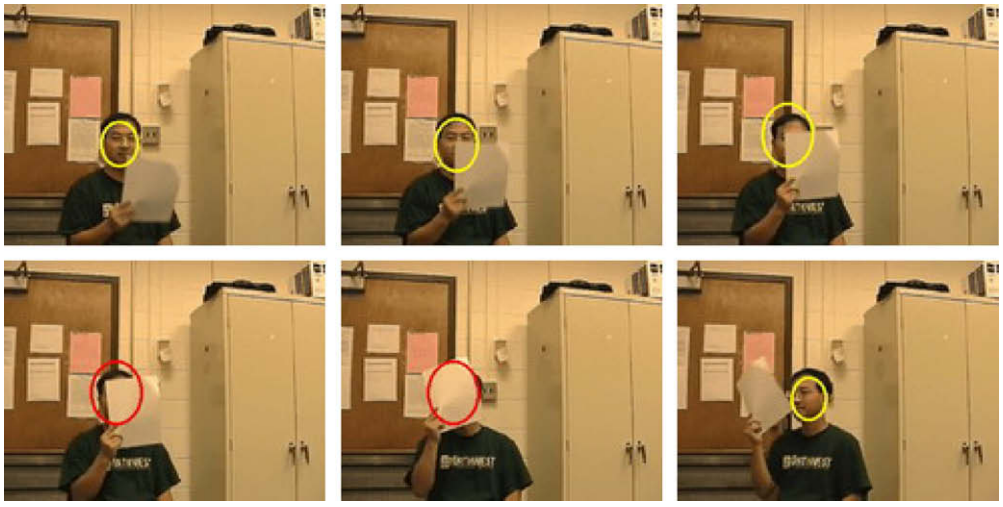

Fig. 11. Tracking results in the presence of occlusions on test video 1. The yellow ellipse implies that no occlusion has occurred, whereas the red ellipse implies that occlusion has occurred. From top left to bottom right, the frame numbers are 356, 359, 362, 366, 382 and 397.

Fig. 12. Tracking results in the presence of two faces on test video 2. The yellow ellipse implies that no occlusion has occurred, whereas the red ellipse implies that occlusion has occurred. From top left to bottom right, the frame numbers are 2, 4, 25, 38, 78 and 138.

Fig. 13. Tracking results with the BAPF at six different times in test video 3. From top left to bottom right, the frame numbers are 12, 40, 61, 136, 158 and 180.

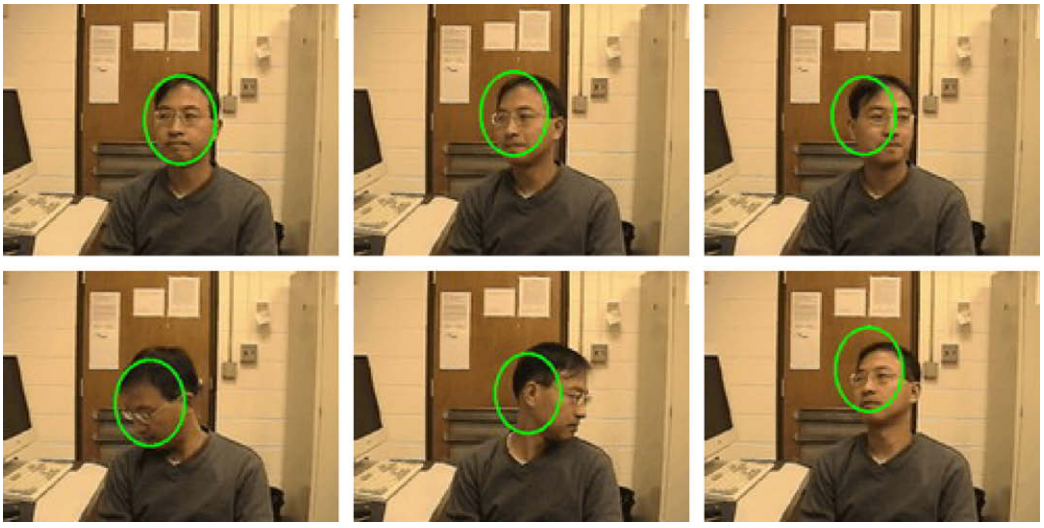

Fig. 14. Tracking results with the Condensation algorithm at same times as in Fig. 14. From top left to bottom right, the frame numbers are 12, 40, 61, 136, 158 and 180.

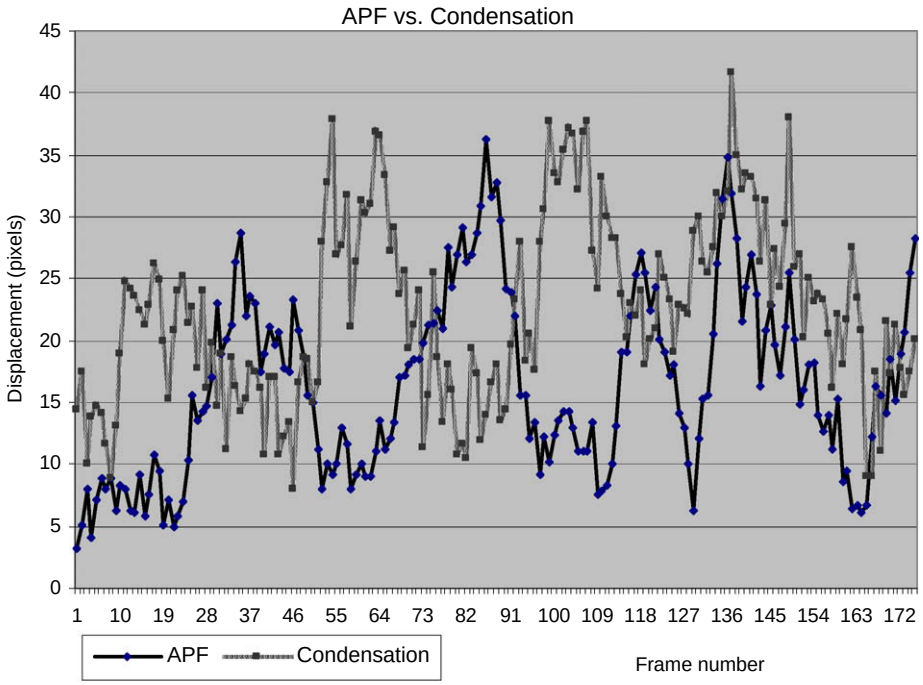

Fig. 15. Tracking results for the APF and Condensation algorithms.

Table 1
Summary of tracking results for the APF and Condensation algorithms

	APF	Condensation
Mean displacement error (pixels)	16.3	22.4
Standard deviation (pixels)	7.3	7.3
Speed (frames/s)	4.7	6.8

iteration than does the Condensation algorithm in order to yield more accurate estimates of the proposal distribution and the posterior distribution. In summary, the APF algorithm can be seen to offer significantly more accurate tracking than the conventional particle filter, albeit at the cost of a moderate though tolerable loss in tracking speed.

Next, the performance of the BAPF algorithm is compared to that of the APF algorithm. Both algorithms employ $N = 1000$ particles for face tracking in test video 3. The upper bound on the number of iterations of the loop controlled by parameter l is $L = 3$ for both algorithms. In the BAPF algorithm, the weight assigned to the result of AdaBoost face detection is $\gamma = 0.8$. The number of successive frames F over which the output of the AdaBoost face detection algorithm is averaged before combining it with the APF output is $F = 1$. The experimental results, shown in Fig. 16 and Table 2, demonstrate that the tracking accuracy of the BAPF algorithm is higher than that of the APF algorithm. It can be seen from Table 1 that the mean displacement error in the case of the BAPF algorithm is significantly lower than that in the case of the APF algorithm. However, the tracking speed of the BAPF algorithm is slightly slower than that of the APF algorithm since the BAPF algorithm runs the AdaBoost face detection procedure on each input frame of the video sequence. Thus, the BAPF algorithm can be seen to significantly improve the tracking accuracy of the APF algorithm, albeit at the cost of a slight decrease in tracking speed compared to the APF algorithm.

The performance of the APF algorithm is analyzed using different values of the parameter L which is the upper bound on the number of iterations of the loop controlled by the variable l in the APF algorithm (Fig. 2). The APF algorithm employs $N = 1000$ particles for face tracking in the test video 3. The value of L is varied from $L = 1$ to $L = 4$ in the experiments. The experimental results, as shown in Fig. 17 and Table 3, demonstrate that the tracking accuracy of the APF algorithm improves as the value of L increases. It can be seen from Table 3 that the mean displacement error in the case of the APF algorithm decreases with increasing values of L. Thus, the APF algorithm with a larger value of L provides more accurate tracking. However, the tracking speed of the APF algorithm also decreases with increasing values of L since the APF algorithm performs more computation per iteration to estimate the posterior distribution. From Fig. 17 and Table 3, it can be seen that as the value of L is increased beyond 3, the accuracy of estimation of the posterior distribution improves only slightly whereas the computation time increases significantly. In order to strike a reasonable balance between tracking accuracy and tracking speed for real-time applications, we choose $L = 3$ for all of our experiments with the APF algorithm and the BAPF algorithm.

The performance of the BAPF algorithm is also analyzed for different values of the parameter F. In Eq. (19), we combine the results of the APF algorithm and the AdaBoost algorithm to obtain the current contour of the tracked face in the current frame. The AdaBoost face detection procedure is performed for each frame of the input video sequence. Using the AdaBoost face detection algorithm, we obtain the estimated position of a detected face by averaging the output of the AdaBoost face detection algorithm over F successive frames including the current frame (i.e., the current frame and $F - 1$ previous frames). For example, for $F = 3$, we average the positions of the detected face in the current frame and the previous two frames (using the AdaBoost face detection algorithm) to obtain an estimated position of the face in the current frame, which is then combined with the output of the APF algorithm using Eq. (19).

In this experiment, the weight assigned to the result of AdaBoost face detection procedure in the BAPF algorithm is $\gamma = 0.8$. The value of F is varied within a set of values $S_F = \{1, 3, 5, 10\}$ whereas the upper bound on the number of sampling iterations L is fixed at 3. The BAPF algorithm employs $N = 1000$ particles for

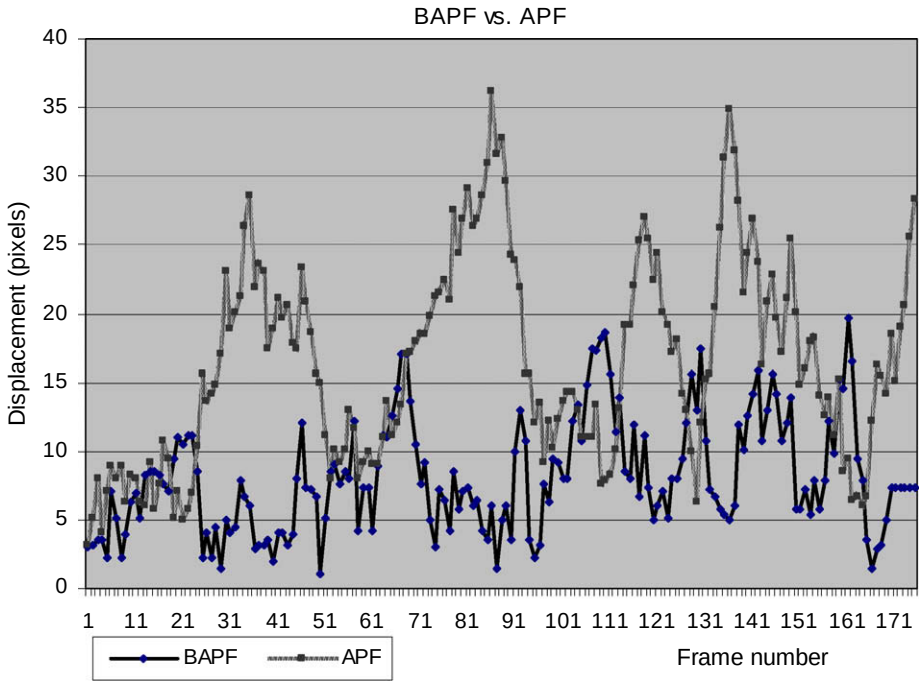

Fig. 16. Tracking results of the BAPF algorithm and the APF algorithm.

Table 2
Summary of tracking results of the BAPF and the APF algorithms

	BAPF	APF
Mean displacement error (pixels)	8.1	16.3
Standard deviation (pixels)	4.1	7.3
Speed (frames/s)	4.1	4.7

Table 3
Summary of tracking results for the APF algorithm for different values of L

	$L = 4$	$L = 3$	$L = 2$	$L = 1$
Mean displacement error (pixels)	16.0	16.3	17.2	22.4
Standard deviation (pixels)	8.7	7.3	9.6	7.3
Speed (frames/s)	3.2	4.7	5.8	6.8

face tracking in test video 3. The experimental results in Fig. 18 and Table 4, show that tracking accuracy of the BAPF algorithm decreases with increasing values of F. As can be seen from Table 4, the mean displacement error in BAPF algorithm is an increasing

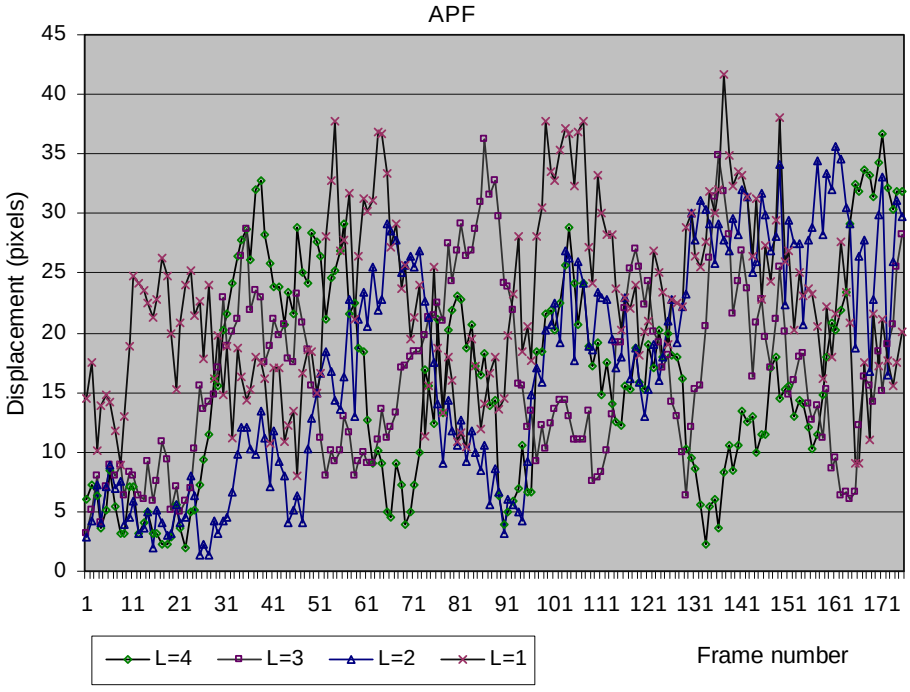

Fig. 17. Tracking results of the APF algorithm for different values of L.

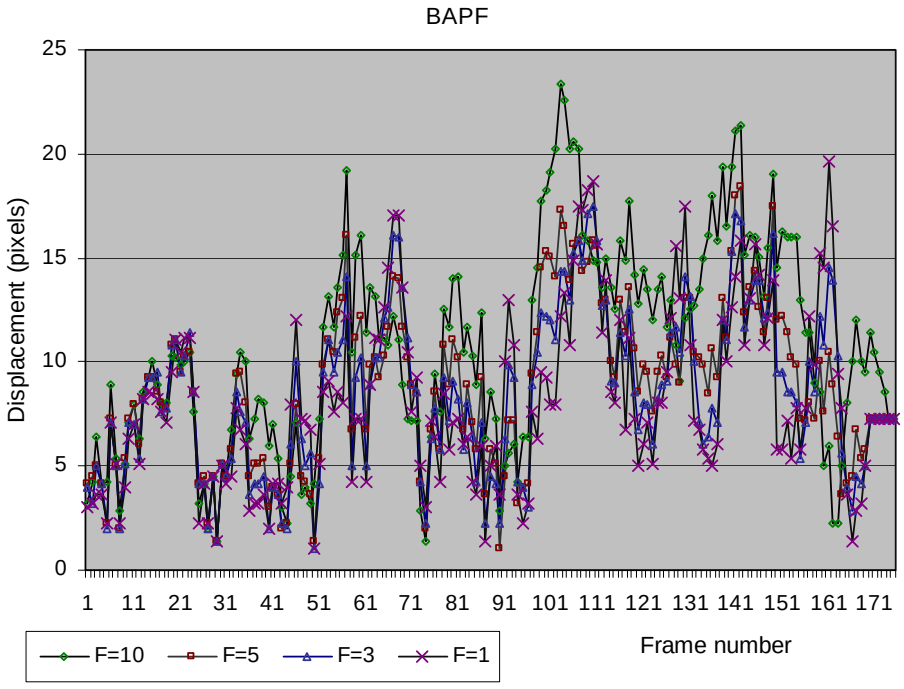

Fig. 18. Tracking results of the BAPF algorithm for different values of *F*.

Table 4
Summary of tracking results of the BAPF for different values of *F*

	F = 10	*F* = 5	*F* = 3	*F* = 1
Mean displacement error (pixels)	10.6	8.8	8.4	8.1
Standard deviation (pixels)	4.9	3.8	3.8	4.1
Speed (frames/s)	4.1	4.1	4.1	4.1

function of *F*. The tracking speed of the BAPF algorithm does not depend on the value of the parameter *F*. Hence, we choose $F = 1$ for the BAPF algorithm in our experiments, i.e., we consider the result of the AdaBoost face detection procedure only for the current frame before integrating the results of the AdaBoost face detection procedure and the APF algorithm.

The performance of the BAPF algorithm is analyzed using different values of the parameter γ (denoted as gamma in Fig. 19), where γ is a weight assigned to the result of the AdaBoost face detection procedure in the BAPF algorithm as shown in Eq. (19). The weight γ is varied within a set of values consisting of $S\gamma = \{0,\ 0.5,\ 0.8,\ 1.0\}$. When $\gamma = 0$, the BAPF algorithm is equivalent to the pure APF algorithm. By increasing the value of γ, we lay greater emphasis on the result of the AdaBoost face detection procedure. When $\gamma = 1$, the BAPF algorithm is equivalent to the pure AdaBoost face detection algorithm. The BAPF algorithm employs $N = 1000$ particles for face tracking in the test video 3 with parameters $F = 1$ and $L = 3$. The experimental results in Fig. 19 and Table 5, show that tracking accuracy of the BAPF algorithm varies significantly with different values of the parameter γ. It can be seen from Table 5 that the mean displacement error in the case of the BAPF algorithm with $\gamma = 0.8$ is the least, and with $\gamma = 0$ is the highest. Thus the experimental results show that the performance of the BAPF-based tracker that simply uses the AdaBoost face detection algorithm in every single frame is the worst. This can be attributed to the fact that the AdaBoost face detection algorithm by itself is prone to detection of false faces or missing the tracked faces in the video sequence. In the case of the BAPF algorithm where $0 < \gamma < 1$, the APF tracking algorithm provides regions of interest to the AdaBoost face detection algorithm, whereas the AdaBoost face detection algorithm, in turn, provides a means for initialization and verification of the output of the APF tracking algorithm via the combination function in Eq. (19). Consequently, on account of the synergetic interaction between the APF tracking algorithm and the AdaBoost face detection algorithm, the performance of the BAPF algorithm is superior to that of either the APF algorithm or the AdaBoost algorithm used in isolation. Table 5 shows that the tracking speed of the BAPF algorithm does not vary for different values of γ since the APF algorithm and the AdaBoost algorithm are both performed for each frame of the input video sequence. In our experiments, we choose $\gamma = 0.8$ for the BAPF algorithm.

5.2.3. Discussion of BAPF experimental results

The BAPF-based face tracker was observed to be successful in tracking the faces throughout each of the test video sequences except when the face being tracked is completely occluded for a long time duration, as shown in Fig. 20. In this case, the occluded face cannot be distinguished from the foreground or the background using the AdaBoost face detector. In the case where the face is occluded for a short time duration, we assume that the occluded face is stationary over the time period of occlusion. In such an instance, the BAPF algorithm is able to recover from temporary tracking failure since the AdaBoost face detection algorithm is capable of reinitializing the APF tracking algorithm. However, the above assumption does not hold in the case of occlusion that lasts for a longer time period, since the person with the occluded face could well exit the scene during the time period of occlusion. When the faces of three people are occluded and are aligned with the optical axis of the camera as shown in Fig. 20b, it is hard to detect and track the faces of the two people that are farthest from the camera, which results in tracking failure as well. From both the cases shown in Fig. 20, it is clear that from the appearance of the face alone it is not possible to reliably deduce the true location of the occluded face. For more robust and accurate tracking, we need to exploit further sources of information such as the appearance of the body or the limbs to augment the BAPF algorithm in order to handle cases of complete occlusion over long time durations. However, the augmented appearance model for accurate tracking

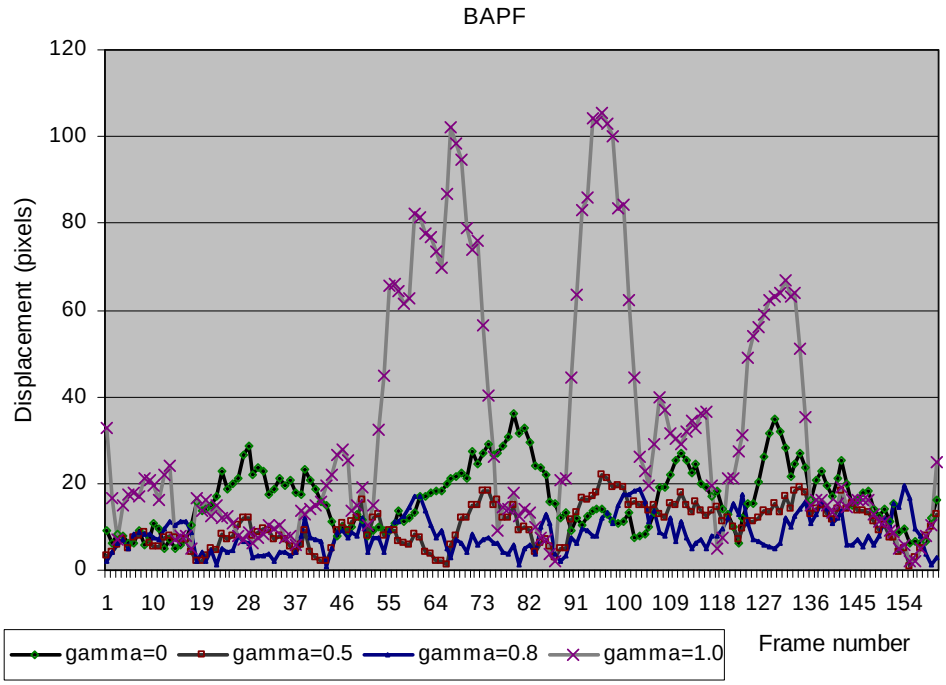

Fig. 19. Tracking results of the BAPF algorithm with different values of the parameter γ.

Table 5
Summary of tracking results of the BAPF with different values of the parameter γ

	$\gamma = 0$	$\gamma = 0.5$	$\gamma = 0.8$	$\gamma = 1.0$
Mean displacement error (pixels)	16.3	10.0	8.1	36.8
Standard deviation (pixels)	7.3	4.7	4.1	32.6
Speed (frames/s)	4.1	4.1	4.1	4.1

would increase the complexity of the dynamical model, which, in turn, may reduce the robustness of the tracker in other scenarios.

6. Conclusions

This paper proposes a novel algorithm for face detection and tracking based on a combination of a novel adaptive particle filtering algorithm and the AdaBoost face detection algorithm. The proposed algorithm provides a general framework for detection and tracking of faces in video sequences. The proposed framework is also applicable to the tracking of other types of objects such as deformable and elastic objects if appropriate contour models such as B-splines are used. The proposed Adaptive Particle Filter (APF) uses a new sampling technique to obtain accurate estimates of the proposal distribution and the posterior distribution for improving tracking accuracy in video sequences. The proposed scheme termed as the Boosted Adaptive Particle Filter (BAPF) combines the APF with the AdaBoost face detection algorithm. The AdaBoost face detection algorithm is used to detect faces in the input images, whereas the APF is used to track the faces in the video sequences. The proposed BAPF algorithm is employed for face detection, face verification, and face tracking in video sequences. It is experimentally shown that the performance of face detection and face tracking can be mutually improved via synergetic interaction in the proposed BAPF scheme resulting in a tracker that is both accurate and computationally efficient.

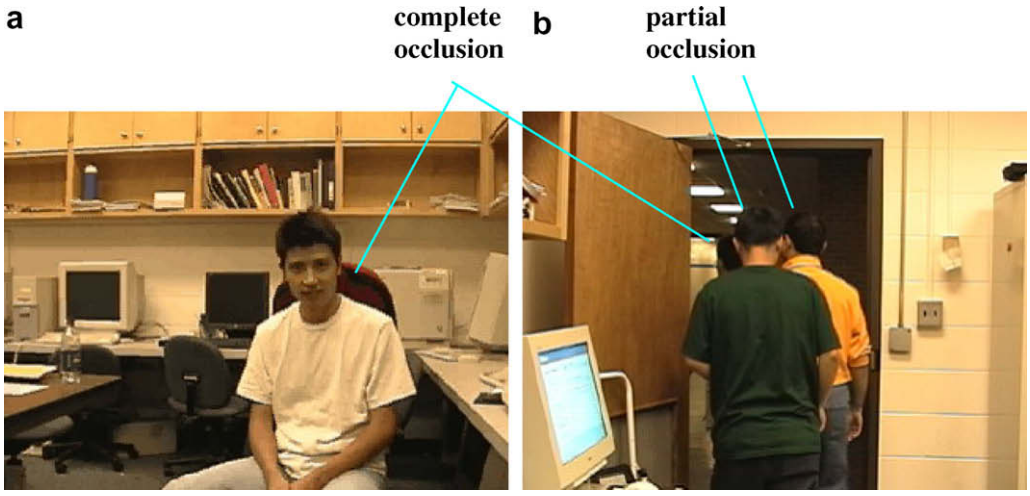

Fig. 20. (a) Tracking failure in case of occlusion for a long time duration. (b) Tracking failure in case of three people overlapping.

The experimental results confirm that the proposed BAPF algorithm provides robust face detection and accurate face tracking in various situations, such as changes in ambient illumination, object scale, object pose and object viewpoint, and instances of occlusion. The performance of the proposed BAPF algorithm is compared to that of the Condensation algorithm, which is a well-known implementation of a general particle filter. The experimental results show that the tracking accuracy of the proposed BAPF algorithm is superior to that of the Condensation algorithm. However, the BAPF algorithm is computationally more intensive than the Condensation algorithm since the BAPF algorithm performs more computation per iteration in order to obtain more accurate non-linear estimates of the proposal and posterior distributions. The experiments also show that the tracking accuracy of the BAPF algorithm is better than that of the APF algorithm, which in turn is better than that of the Condensation algorithm. The proposed BAPF algorithm also overcomes the limitations of the Gaussian distribution assumption in linear filtering systems such as the Kalman filter and its various variants.

The problem of increased computational overhead entailed in the case of the BAPF algorithm can be alleviated to some degree by reducing the number of particles. However, the tracking accuracy typically reduces as the number of particles in a particle filter decrease. In a real-world application, one has strike an appropriate balance between the tracking accuracy and the computational overhead of the chosen tracking algorithm. However, currently there are no published analytical results describing the mathematical relation between the number of particles and the tracking performance for a given tracking application. In future, we hope to further analyze this tradeoff and reduce the computational cost in order to make the BAPF algorithm computationally more efficient. We also expect to improve the tracking performance of the BAPF algorithm by further enhancing the APF algorithm to deal with occlusions that occur very frequently or persist over a long time period.

Appendix A

A.1. Glossary of mathematical notation

$\mathbf{x}$	state vector for an object contour;
$\mathbf{y}$	observation vector;
$p(\mathbf{y}\|\mathbf{x})$	observation likelihood (also termed as observation density);
T	length of the measurement line;
x_i	a finite number of sample points on a contour;
s_i	normal to the contour (also termed as measurement line);
m	index of the detected features;
m_i	number of the detected features;
z_i	edge feature;
$p_T(m_i)$	Poisson distribution of clutter features on the measurement line;
λ	feature density parameter in the Poisson distribution $p_T(m_i)$;
$p_{x_i}(\mathbf{z}\|\mathbf{v}=\{x_i\})$	generic likelihood function of the observation at a sample point x_i ($i = 1,2,\ldots,n$);
q_0	probability of undetected features for an object boundary;
q_1	probability of detected features for an object boundary;
$\mathbf{x}_t$	state vector for an object at time t;
$\mathbf{x}_{1:t} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_t\}$	state history vector up to time t;
$\mathbf{y}_t$	observation vector at time t;
$\mathbf{y}_{1:t} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t\}$	observation history vector up to time t;
$\bar{\mathbf{x}}$	mean value of a state vector;
$\boldsymbol{\omega}_t$	Gaussian noise;
$\mathbf{A}$	matrix describing the deterministic component of the dynamical model;
$\mathbf{B}$	matrix describing the stochastic component of the dynamical model;
$p(\mathbf{x}_t\|\mathbf{x}_{t-1})$	dynamical model (also termed as transition prior);
$p(\mathbf{x}_t\|\mathbf{y}_{1:t})$	posterior density;
$p(\mathbf{x}_t\|\mathbf{y}_{1:t-1})$	effective prior;
$p(\mathbf{y}_t\|\mathbf{y}_{1:t-1})$	observation prior;
$f(\mathbf{x}_t)$	function of object state vector;
$E[f(\mathbf{x}_t)]$	estimate or expectation of function $f(\mathbf{x}_t)$;
L	number of iterations of loop l in the adaptive particle filter (APF);
N	number of particles;
$w_t^{(i)}$	weight of particle i at time t;
$\mathbf{x}_t^{(i)}$	state vector of particle i at time t;
$q(\mathbf{x}_t\|\mathbf{x}_{t-1},\mathbf{y}_{1:t})$	proposal distribution;
$\mathcal{N}(\hat{\mathbf{x}}_t^{(i)}, \hat{\mathbf{P}}_t^{(i)})$	Gaussian distribution;
$\mathbf{x}_{t,l}^{(i)}$	particle state vector computed within loop l;
$u_l(\mathbf{x})$	proposal distribution computed within loop l used in the adaptive particle filter (APF);
$\xi_1^{(i)}$, $\xi_2^{(i)}$	two specific values in domain D
m_1, M_1	two specific values of a continuous function in domain D;
m_2, M_2	two specific values of a continuous function in domain D;
Φ	a continuous function in domain D;
$\max_l$, $\min_l$	two specific values in domain D used to impose bounds on Φ within loop l;
K_l	constant within loop l;
$E[f(\mathbf{x}), \hat{p}(\mathbf{x})]$	sampling error in iteration step l with respect to $f(\mathbf{x})$;
$E_c(f(\mathbf{x}_t))$	estimate of a sampled point on the contour combining the estimated values from the APF and the AdaBoost face detection algorithm;
γ	weight assigned to the output of the Adaboost face detection algorithm;
η	confidence measure for each detected face in the image;
d	distance between the center of a detected face and the center of a sampled template contour;
F	number of the previous frames used for the estimation of an object in the current frame.

A.2. Modeling the Adaptive Learning Constraint

A critical step in the Adaptive Particle Filter (APF) is obtaining a good approximation to the sampling proposal distribution $u_l(\mathbf{x})$, as shown in the sampling step in Fig. 2. The primary purpose of estimating the proposal distribution $u_l(\mathbf{x})$ recursively in a given state is to reduce the estimation error, which is a direct result of approximating the posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ with a finite number of particles. The design of a single iteration of the estimation of the proposal distribution $u_l(\mathbf{x})$ will be presented in the following analysis. The iteration is described by the outer loop controlled by the parameter l: $l = 1, 2, \ldots, L$ in Step 2 of Fig. 2, where L is a preset default value. The additional iterations of loop l in the first loop of step 2 could be used to adaptively reduce the estimation error.

We present the following analysis to prove the superiority of the proposed APF over the conventional particle filter. First, we prove that successive APF iterations result in the asymptotic convergence of the estimate of the proposal distribution. The proof of convergence shows that the estimation error for the proposal distribution at loop step $l = k + 1$ is less than the estimation error of the proposal distribution at loop step $l = k$, where $k \in (1,2,\ldots,L-1)$. This results in a better approximation to the proposal distribution and the posterior distribution via the multiple iterations of loop l. Thus, one can obtain lower estimation errors of the proposal distribution and the posterior distribution. Second, we present a precise definition of the Adaptive Learning Constraint in the following analysis, which serves to clarify the APF algorithm described in Fig. 2.

We define the error of a sampling function $\hat{p}(\mathbf{x})$ with respect to $f(\mathbf{x})$ as

$$E[f(\mathbf{x}),\hat{p}(\mathbf{x})]=\left|\int f(\mathbf{x})(p(\mathbf{x})-\hat{p}(\mathbf{x}))\mathrm{d}\mathbf{x}\right| \tag{A-1}$$

where $f(\mathbf{x})$ represents any continuous function of the state vector $\mathbf{x}$ of the tracked object. In the context of our problem, the state vector $\mathbf{x}$ represents the location of the contour of the tracked object. Also,

$$p(\mathbf{x})=p(\mathbf{x}_t|\mathbf{y}_{1:t})=\frac{p(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{t-1}}{\int p(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{t-1}\mathrm{d}\mathbf{x}_t},$$

$$\hat{p}(\mathbf{x})=\hat{p}(\mathbf{x}_t|\mathbf{y}_{1:t})\approx\sum_{i=1}^{N}w_t^{(i)}\delta(\mathbf{x}_t-\mathbf{x}_t^{(i)}),\ \text{and}\ |\cdot|\ \text{denotes the Euclidean norm.}$$

The propagation of errors between successive iterations of the APF can be analyzed as follows. Specifically, we consider a single iteration step l: $l\in\{1,2,\ldots,L\}$ in Step 2 of the APF algorithm (Fig. 2). From the APF algorithm (Fig. 2), the estimate of the proposal distribution in iteration step l is given by

$$p_l(\mathbf{x})=\sum_{i=1}^{N}w_{t,l}^{(i)}\delta(\mathbf{x}_t-\mathbf{x}_{t,l}^{(i)}) \tag{A-2}$$

Thus, the sampling error $E[f(\mathbf{x}),\hat{p}(\mathbf{x})]$ at iteration step l with respect to $f(\mathrm{x})$ is computed as

$$\begin{aligned}E[f(\mathbf{x}),\hat{p}(\mathbf{x})]&=\left|\int f(\mathbf{x})(p(\mathbf{x})-p_l(\mathbf{x}))\mathrm{d}\mathbf{x}\right|\\&=\left|f(\mathbf{x}_t)\left(\frac{p(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{t-1}}{\int p(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{t-1}\mathrm{d}\mathbf{x}_t}\right.\right.\\&\quad\left.\left.-\sum_{i=1}^{N}w_{t,l}^{(i)}\delta(\mathbf{x}_t-\mathbf{x}_{t,l}^{(i)})\right)\mathrm{d}\mathbf{x}_t\right|\end{aligned} \tag{A-3}$$

Based upon Eq. (19), we have $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})=\sum_{i=1}^{N}w_{t-1}^{i}\delta(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{i})$. Substituting $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ with $\sum_{i=1}^{N}w_{t-1}^{i}\delta(\mathbf{x}_{t-1}-\mathbf{x}_{t-1}^{i})$ in Eq. (A-3) and performing the Dirac function computation, we obtain the estimation error as

$$\begin{aligned}E[f(\mathbf{x}),p_l(\mathbf{x})]&=\left|f(\mathbf{x}_t)\left(\frac{p(\mathbf{y}_t|\mathbf{x}_t)\sum_{i=1}^{N}w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{\int p(\mathbf{y}_t|\mathbf{x}_t)\sum_{i=1}^{N}w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})\mathrm{d}\mathbf{x}_t}\right.\right.\\&\quad\left.\left.-\sum_{i=1}^{N}w_{t,l}^{(i)}\delta(\mathbf{x}_t-\mathbf{x}_{t,l}^{(i)})\right)\mathrm{d}\mathbf{x}_t\right|=\left|f(\mathbf{x}_t)\right.\\&\quad\left(\frac{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_t)w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{\sum_{i=1}^{N}\int p(\mathbf{y}_t|\mathbf{x}_t)w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})\mathrm{d}\mathbf{x}_t}\right.\\&\quad\left.\left.-\sum_{i=1}^{N}w_{t,l}^{(i)}\delta(\mathbf{x}_t-\mathbf{x}_{t,l}^{(i)})\right)\mathrm{d}\mathbf{x}_t\right|\end{aligned} \tag{A-4}$$

From the APF algorithm (Fig. 2), we know that

$$w_{t,l}^{(i)}=p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)} \tag{A- 5}$$

$$w_{t,l}^{(i)}=\frac{w_{t,l}^{(i)}}{\sum_{i=1}^{N}w_{t,l}^{(i)}}=\frac{p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}} \tag{A-6}$$

After combining Eq. (A-4), Eq. (A-5), and Eq. (A-6), we obtain

$$\begin{aligned}E[f(\mathbf{x}),p_l(\mathbf{x})]&=\left|f(\mathbf{x}_t)\left(\frac{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_t)w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{\sum_{i=1}^{N}\int p(\mathbf{y}_t|\mathbf{x}_t)w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})\mathrm{d}\mathbf{x}_t}\right.\right.\\&\quad\left.\left.-\sum_{i=1}^{N}\frac{p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}\delta(\mathbf{x}_t-\mathbf{x}_{t,l}^{(i)})\right)\mathrm{d}\mathbf{x}_t\right|\\&=\left|\int\left(\frac{f(\mathbf{x}_t)\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_t)w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})}{\sum_{i=1}^{N}\int p(\mathbf{y}_t|\mathbf{x}_t)w_{t-1}^{(i)}p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})\mathrm{d}\mathbf{x}_t}\right)\mathrm{d}\mathbf{x}_t\right.\\&\quad\left.-\frac{\sum_{i=1}^{N}f(\mathbf{x}_{t,l}^{(i)})p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}\right|\\&=\left|\frac{\sum_{i=1}^{N}\int f(\mathbf{x}_t)p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}\mathrm{d}\mathbf{x}_t}{\sum_{i=1}^{N}\int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}\mathrm{d}\mathbf{x}_t}\right.\\&\quad\left.-\frac{\sum_{i=1}^{N}f(\mathbf{x}_{t,l}^{(i)})p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}\right|\end{aligned} \tag{A-7}$$

Using the Lagrange theorem, we could obtain specific values $\xi_1^{(i)}$ and $\xi_2^{(i)}$ in domain D: $\xi_1^{(i)}\in D,\ \xi_2^{(i)}\in D,\ (i=1,2,3,\ldots,N)$ such that

$$\begin{aligned}&\int f(\mathbf{x}_t)p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}\mathrm{d}\mathbf{x}_t\\&\quad=f(\xi_1^{(i)})p(\mathbf{y}_t|\xi_1^{(i)})p(\xi_1^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}\end{aligned} \tag{A-8}$$

$$\int p(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}\mathrm{d}\mathbf{x}_t=p(\mathbf{y}_t|\xi_2^{(i)})p(\xi_2^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)} \tag{A-9}$$

$\xi_1^{(i)},\ \xi_2^{(i)}$ can be obtained by searching in the domain D.

Therefore we obtain

$$\begin{aligned}E[f(\mathbf{x}),p_l(\mathbf{x})]&=\left|\frac{\sum_{i=1}^{N}f(\xi_1^{(i)})p(\mathbf{y}_t|\xi_1^{(i)})p(\xi_1^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\xi_2^{(i)})p(\xi_2^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}}\right.\\&\quad\left.-\frac{\sum_{i=1}^{N}f(\mathbf{x}_{t,l}^{(i)})p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}\right|\\&=\sum_{i=1}^{N}\left|\frac{f(\xi_1^{(i)})p(\mathbf{y}_t|\xi_1^{(i)})p(\xi_1^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\xi_2^{(i)})p(\xi_2^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}}\right.\\&\quad\left.-\frac{f(\mathbf{x}_{t,l}^{(i)})p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}{\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}}\right|\end{aligned} \tag{A-10}$$

Suppose that $f(\mathbf{x})$, $p(\mathbf{y}_t|\mathbf{x}_t)$, $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$ are continuous functions on domain D, hence we have the following equation

$\exists m_1,\ M_1\in R$, such that

$$\begin{aligned}&m_1\left|\xi_1^{(i)}-\mathbf{x}_{t,l}^{(i)}\right|\\&\le\left|f(\xi_1^{(i)})p(\mathbf{y}_t|\xi_1^{(i)})p(\xi_1^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}-f(\mathbf{x}_{t,l}^{(i)})p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}\right|\\&\le M_1\left|\xi_1^{(i)}-\mathbf{x}_{t,l}^{(i)}\right|\end{aligned} \tag{A- 11}$$

Likewise, we have

$\exists m_2,\ M_2\in R$, such that

$$\begin{aligned}&m_2\left|\xi_2^{(i)}-\mathbf{x}_{t,l}^{(i)}\right|\\&\le\left|\sum_{i=1}^{N}p(\mathbf{y}_t|\xi_2^{(i)})p(\xi_2^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)}-\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}\right|\\&\le M_2|\xi_2^{(i)}-\mathbf{x}_{t,l}^{(i)}|\end{aligned} \tag{A- 12}$$

Let

$$F_1^{(i)}=f(\xi_1^{(i)})p(\mathbf{y}_t|\xi_1^{(i)})p(\xi_1^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)} \tag{A- 13}$$

$$F_2=\sum_{i=1}^{N}p(\mathbf{y}_t|\xi_2^{(i)})p(\xi_2^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t-1}^{(i)} \tag{A- 14}$$

$$\Delta F_1^{(i)}=f(\mathbf{x}_{t,l}^{(i)})p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}-F_1^{(i)} \tag{A- 15}$$

$$\Delta F_2=\sum_{i=1}^{N}p(\mathbf{y}_t|\mathbf{x}_{t,l}^{(i)})p(\mathbf{x}_{t,l}^{(i)}|\mathbf{x}_{t-1}^{(i)})w_{t,l-1}^{(i)}-F_2 \tag{A-16}$$

Thus we obtain

$$E[f(\mathbf{x}), p_l(\mathbf{x})] = \sum_{i=1}^{N} \left| \frac{F_1^{(i)}}{F_2} - \frac{F_1^{(i)} + \Delta F_1^{(i)}}{F_2 + \Delta F_2} \right| = \sum_{i=1}^{N} \left| \frac{F_1^{(i)}(F_2 + \Delta F_2) - F_2(F_1^{(i)} + \Delta F_1^{(i)})}{F_2(F_2 + \Delta F_2)} \right| = \sum_{i=1}^{N} \left| \frac{F_1^{(i)} \Delta F_2 - F_2 \Delta F_1^{(i)}}{F_2(F_2 + \Delta F_2)} \right| \approx \sum_{i=1}^{N} \left| \frac{F_1^{(i)} \Delta F_2 - F_2 \Delta F_1^{(i)}}{F_2^2} \right| \quad \text{(A-17)}$$

Let

$$\Phi = \sum_{i=1}^{N} \left| F_1^{(i)} \Delta F_2 - F_2 \Delta F_1^{(i)} \right| \quad \text{(A-18)}$$

Since $f(\mathbf{x})$, $p(\mathbf{y}_t|\mathbf{x}_t)$, $p(\mathbf{x}_t|\mathbf{x}_{t-1}^{(i)})$ are continuous functions defined on domain D, $F_1^{(i)}$, F_2, $\Delta F_1^{(i)}$, ΔF_2 are also continuous functions on domain D. Furthermore, Φ is also a continuous function on domain D. Based on the properties of continuous functions, Eq. (A-11) and Eq. (A-12), Φ is bounded by two specific values, $\max_l$ and $\min_l$.

Let

$$\max_l = \max_{1 \le i \le N} \{M_1 |\xi_1^{(i)} - \mathbf{x}_{t,l}^{(i)}|, M_2 |\xi_2^{(i)} - \mathbf{x}_{t,l}^{(i)}|\} \quad \text{(A- 19)}$$

$$\min_l = \min_{1 \le i \le N} \{m_1 |\xi_1^{(i)} - \mathbf{x}_{t,l}^{(i)}|, m_2 |\xi_2^{(i)} - \mathbf{x}_{t,l}^{(i)}|\} \quad \text{(A-20)}$$

where the values of M_1, M_2, m_1 and m_2 can be obtained by performing an exhaustive search in the domain D.

We obtain

$$K_l \cdot \min_l \le E(f(X), p_l(X)) \le K_l \cdot \max_l, \quad \text{for loop step } l \quad \text{(A-21)}$$

where K_l is a constant.

Likewise, we can get the following equation at loop step $l-1$:

$$K_{l-1} \cdot \min_{l-1} \le E(f(\mathbf{x}), p_{l-1}(\mathbf{x})) \le K_{l-1} \cdot \max_{l-1}, \quad \text{for loop step } l-1 \quad \text{(A-22)}$$

Let

$$\frac{K_l \cdot \max_l}{K_{l-1} \cdot \min_{l-1}} \le \alpha < 1, \quad \text{where } 0 < \alpha < 1. \quad \text{(A-23)}$$

Thus we obtain

$$E(f(X), p_l(X)) \le \alpha \cdot E(f(X), p_{l-1}(X)), \quad 0 < \alpha < 1 \quad \text{(A-24)}$$

The value of α is obtained from Eq. (A-23), which determines the convergence. The smaller the value of α, the faster the convergence of the algorithm.

If Eq. (A-23) is satisfied, then Eq. (A-24) ensures that the estimation error for the proposal distribution and posterior distribution converges over successive iterations. The inequality in Eq. (A-23) is only a necessary condition for the convergence of the estimate, the parameters of which can be learned from the computations performed during the iterations of the APF. Hence it is termed the Adaptive Learning Constraint (ALC). The inequality in Eq. (A-23) can be alternatively represented as

$$K_l \cdot \max_l \le \alpha \cdot K_{l-1} \cdot \min_{l-1} \quad \text{(A-25)}$$

Either Eq. (A-23) or Eq. (A-25) can be used to represent the ALC.

The satisfaction of the ALC can be guaranteed by searching for the values of $\max_l$ and $\min_{l-1}$ from among the N particles in each iteration as follows

(1) Determine $\xi_1^{(i)}$, $\xi_2^{(i)}$, M_1, M_2, m_1, m_2, ($i = 1, 2, \ldots, N$) from the N particles in loop l.

Determine $\xi_1^{(i)}$, $\xi_2^{(i)}$, M_1, M_2, m_1, m_2, ($i = 1, 2, \ldots, N$) from the N particles in loop $l-1$.

(2) Perform an exhaustive search for maxl and $\min_{l-1}$ using Eqs. (A-19) and (A-20).

(3) Determine whether or not the ALC is satisfied according to Eq. (A-25).

Thus, we prove that the iterations of loop l result in the convergence of the estimate of the proposal distribution. The proof of convergence also demonstrates that the estimation error of the proposal distribution at loop step $l = k + 1$ is less than the estimation error of the proposal distribution at loop step $l = k$, where $k \in (1, 2, \ldots, L-1)$. This results in a better approximation of the proposal distribution and the posterior distribution via the iterations of the loop controlled by the parameter l. Thus, we obtain a lower estimation error for the proposal distribution and the posterior distribution over the course of the tracking procedure. In summary, we confirm that the APF algorithm with the incorporation of the ALC can result in a more accurate estimate of the proposal distribution and posterior distribution. Generally speaking, as more frames are processed over the course of the tracking procedure, conventional particle filters result in monotonically increasing tracking error. In contrast, the proposed APF algorithm is designed to improve the estimate of the proposal distribution and the posterior distribution as the tracking procedure evolves over time.

References

[1] P. Viola, M. Jones, Robust real time object detection, in: Proc. IEEE ICCV Workshop on Statistical and Computational Theories of Vision, 2001.
[2] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2001.
[3] S.Z. Li, Z.Q. Zhang, H.Y. Shum, H. Zhang, FloatBoost learning and statistical face detection, IEEE Trans. Pattern Anal. Mach. Intell. 26 (9) (2002) 1112–1123.
[4] J.L. Jiang, K. Loe, S-AdaBoost and pattern detection in complex environment, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition Workshops, 2003.
[5] H. Rowley, S. Baluja, T. Kanade, Neural network-based face detection, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, 1996, pp. 203–208.
[6] K. Curran, X. Li, N. McCaughley, Neural network face detection, Imaging Sci. J. 53 (2) (2005) 105–115.
[7] E. Osuna, R. Freund, F. Girosi, Training support vector machines: an application to face detection, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, 1997, pp. 130–136.
[8] P. Shih, C. Liu, Face detection using discriminating feature analysis and support vector machine in video, in: Proc. 17th Intl. Conf. Pattern Recognition, 2004.
[9] L.R. Rabiner, B.H. Jung, Fundamentals of Speech Recognition, Prentice Hall, 1993.
[10] H. Schneiderman, T. Kanade, Probabilistic modeling of local appearance and spatial relationships for object recognition, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, 1998, pp. 45–51.
[11] H. Schneiderman, A statistical approach to 3D object detection applied to faces and cars, Ph.D. Dissertation, R.I., Carnegie Mellon University, 2000.
[12] A.J. Davison, D.W. Murray, Mobile robot localisation using active vision, in: Proc. 5th European Conf. Computer Vision, Freiburg, Germany, 1998.
[13] M. Borg, D. Thirde, J. Ferryman, F. Fusier, V. Valentin, F. Brémond, M. Thonnat, Video surveillance for aircraft activity monitoring, in: IEEE Intl. Conf. Advanced Video and Signal-based Surveillance, Como, Italy, 2005, pp. 17–21.
[14] D.W. Hansen, R. Hammoud, Boosting particle filter-based eye tracker performance through adapted likelihood function to reflexions and light changes, in: IEEE Intl. Conf. Advanced Video and Signal-based Surveillance, Como, Italy, 2005, pp. 111–117.
[15] K. Nummiaro, E. Koller-Meier, L.V. Gool, An adaptive color-based particle filter, Image Vis. Comput. 21 (1) (2003) 99–110.
[16] S. Intille, J. Davis, A. Bobick, Real-time closed-world tracking, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, 1997, pp. 697–703.
[17] A. Blake, M. Isard, Active Contours, Springer-Verlag, London, 1998.
[18] J. MacCormick, Probabilistic models and stochastic algorithms of visual tracking, Ph.D. Thesis, University of Oxford, Oxford, UK, 2000.
[19] Y. Rathi, N. Vaswani, A. Tannenbaum, A. Yezzi, Particle filtering for geometric active contours with application to tracking moving and deforming objects, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, 2005.
[20] J. Rehg, T. Kanade, Visual tracking of high dof articulated structures: an application to human hand tracking, in: Proc. 3rd European Conf. Computer Vision, Springer-Verlag, 1994, pp. 35–46.
[21] T. Jebara, K. Russell, A. Pentland, Mixtures of eigenfeatures for real-time structure from texture, in: Proc. IEEE Intl. Conf. Computer Vision, Mumbai, India, 1998, pp. 128–135.
[22] C. Chang, R. Ansari, A. Khokhar, Multiple object tracking with kernel particle filter, in: Proc. IEEE Conf. on Computer Vision and Pattern Recognition, 2005.
[23] P. Li, T. Zhang, A.E.C. Pece, Visual contour tracking based on particle filters, Image Vis. Comput. 21 (2003) 111–123.

[24] K. Okuma, A. Taleghni, N.D. Freitas, J.J. Little, D.G. Lowe, A boosted particle filter: multitarget detection and tracking, in: Proc. European Conf. on Computer Vision, LNCS 3021, 2004, pp. 28–39.
[25] M. Isard, Visual motion analysis by probabilistic propagation of conditional density, Ph.D. Thesis, University of Oxford, Oxford, UK, 1998.
[26] H.L. Wang, L.F. Cheong, MRF augmented particle filter tracker, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, San Diego, CA, 2005.
[27] M. Isard, J. MacCormick, BraMBLe: A Bayesian multiple-blob tracker, in: Proc. IEEE Intl. Conf. Computer Vision, Vancouver, Canada, vol. 2, 2001, pp. 34–41.
[28] J. Vermaak, A. Doucet, P. Perez, Maintaining multi-modality through mixture tracking, in: Proc. IEEE Intl. Conf. Computer Vision, 2003.
[29] A. Blake, M. Isard, D. Reynard, Learning to track the visual motion of contours, Artif. Intell. 78 (1995) 101–134.
[30] J. MacCormick, A. Blake, A probabilistic contour discriminant for object localization, in: Proc. IEEE Intl. Conf. Computer Vision, 1998, pp. 390–395.
[31] H. Luokepohl, Introduction to Multiple Time Series Analysis, second ed., Spring-Verlag, 1993.
[32] A. Blake, R. Curwen, A. Zisserman, A framework for spatio-temporal control in the tracking of visual contours, Intl. J. Comput. Vis. 11 (2) (1993) 127–145.
[33] M. Isard, A. Blake, Condensation-conditional density propagation for visual tracking, Int. J. Comput. Vis. 29 (1) (1998) 5–28.
[34] D. Reynard, A. Wildenberg, A. Blake, J. Marchant, Learning dynamics of complex motions from image sequences, in: Proc. European Conf. Computer Vision, Cambridge, England, 1996, pp. 357–368.
[35] A. Papoulis, Probability and Statistics, Prentice-Hall, 1990.
[36] A. Doucet, N.J. Gordon, V. Krishnamurthy, Particle filters for state estimation of jump Markov linear systems, IEEE Trans. Signal Process. 49 (3) (2001) 613–624.
[37] X. Luo, S. Bhandarkar, Multiple object tracking using elastic matching, in: IEEE Intl. Conf. Advanced Video and Signal based Surveillance, Como, Italy, 2005, pp. 123–128.
[38] D. Koller, J.W. Weber, J. Malik, Robust multiple car tracking with occlusion reasoning, in: European Conf. Computer Vision, Stockholm, Sweden, 1994, pp. 189–196.
[39] D. Terzopoulos, R. Szeliski, Tracking with Kalman snakes, in: Active Vision, MIT Press, Cambridge, USA, 1993, pp. 3–20.
[40] S.J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, H. Wechsler, Tracking groups of people, Comput. Vis. Image Understand. 80 (2000) 42–56.
[41] W.J. Rucklidge, Locating objects using the Hausdorff distance, in: Proc. Intl. Conf. Computer Vision, Massachusetts, USA, 1995, pp. 457–464.
[42] S. Malik, G.Roth, C. McDonald, Robust corner tracking for real-time augmented reality, in: Proc. Vision Interface, Calgary, Canada, 2002, pp. 399–406.
[43] V. Lepetit, J. Pilet, P. Fua, Point matching as a classification problem for fast and robust object pose estimation, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, Washington, DC, USA, vol. 2, 2004, pp. 244–250.
[44] K. Tieu, P. Viola, Boosting image retrieval, in: Proc. IEEE Conf. Computer Vision and Pattern Recognition, vol. 1, 2000, pp. 228–235.
[45] R. Lienhart, J. Maydt, An extended set of Haar-like features for rapid object detection, in: Proc. IEEE Intl. Conf. Image Processing, vol. 1, 2002, pp. 900–903.