Robust Multi-Speaker Tracking via Dictionary Learning and Identity Modeling

Mark Barnard, Peter Koniusz, *Member, IEEE*, Wenwu Wang, *Senior Member, IEEE*, Josef Kittler, *Life Member, IEEE*, Syed Mohsen Naqvi, *Member, IEEE*, and Jonathon Chambers, *Fellow, IEEE*

***Abstract*—We investigate the problem of visual tracking of multiple human speakers in an office environment. In particular, we propose novel solutions to the following challenges: (1) robust and computationally efficient modeling and classification of the changing appearance of the speakers in a variety of different lighting conditions and camera resolutions; (2) dealing with full or partial occlusions when multiple speakers cross or come into very close proximity; (3) automatic initialization of the trackers, or re-initialization when the trackers have lost lock caused by e.g. the limited camera views. First, we develop new algorithms for appearance modeling of the moving speakers based on dictionary learning (DL), using an off-line training process. In the tracking phase, the histograms (coding coefficients) of the image patches derived from the learned dictionaries are used to generate the likelihood functions based on Support Vector Machine (SVM) classification. This likelihood function is then used in the measurement step of the classical particle filtering (PF) algorithm. To improve the computational efficiency of generating the histograms, a soft voting technique based on approximate Locality-constrained Soft Assignment (LcSA) is proposed to reduce the number of dictionary atoms (codewords) used for histogram encoding. Second, an adaptive identity model is proposed to track multiple speakers whilst dealing with occlusions. This model is updated online using Maximum a Posteriori (MAP) adaptation, where we control the adaptation rate using the spatial relationship between the subjects. Third, to enable automatic initialization of the visual trackers, we exploit audio information, the Direction of Arrival (DOA) angle, derived from microphone array recordings. Such information provides, *a priori*, the number of speakers and constrains the search space for the speaker's faces. The proposed system is tested on a number of sequences from three publicly available and challenging data corpora (AV16.3, EPFL pedestrian data set and CLEAR) with up to five moving subjects.**

***Index Terms*—Visual Tracking, Particle Filters, Dictionary Learning.**

Manuscript received March 27, 2013; revised August 19, 2013; accepted November 27, 2013. Date of publication January 22, 2014; date of current version March 13, 2014. This work was supported by the Engineering and Physical Sciences Research Council of the UK (grant no. EP/H050000/1). The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Sen-Ching Cheung.

M. Barnard, W. Wang, and J. Kittler are with the Centre for Vision, Speech and Signal Processing, University of Surrey, Surrey GU2 7XH, U.K. (e-mail: mark.barnard@surrey.ac.uk; w.wang@surrey.ac.uk; j.kittler@surrey.ac.uk).

P. Koniusz was with the Centre for Vision, Speech and Signal Processing, University of Surrey, and is now with INRIA LEAR, Rhône-Alpes 38334, France (e-mail: peter.koniusz@inria.fr).

S. M. Naqvi and J. Chambers are with the Advanced Signal Processing Group, Loughborough University, Leicestershire LE11 3TU, U.K. (e-mail: s.m.r.naqvi@lboro.ac.uk; j.a.chambers@lboro.ac.uk).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2014.2301977

I. Introduction

THE problem of object tracking in computer vision has received much interest from researchers in recent years. Tracking concerns estimating the position of an object, be it a human, an animal, a car or a missile in space and time. Applications for tracking objects are wide ranging, including as diverse sectors as security, defence, robotics, sport, wildlife preservation, as well as communications. In this paper, we focus on the tracking of multiple moving speakers in various indoor environments using multiple cameras with the assistance from microphones. The methods developed here are however not confined to this specific application, and they can be readily applied to track different objects in a variety of environments. There are several fundamental problems to be solved to enable efficient and accurate object tracking. Firstly, we must address the problem of robustly modeling the appearance of the object, due to, for example, the changes in illumination and also object orientation. Secondly, when tracking multiple objects occlusions or near occlusions between objects are a major problem causing loss of tracking of one or both objects, especially when the objects are of similar appearance. Another challenge, particularly in the task of tracking people in meeting room applications, is to preserve the identity of the subjects through occlusions. Lastly, a problem facing most tracking systems is the initialization of object's positions at the start of the tracking sequence. Many systems overcome this by simply manually initializing the object's location before tracking.

There are broadly two approaches to the problem of robust appearance modeling, using either adaptive models or static models. Adaptive models are updated as the object's appearance changes over time and have been used extensively in tracking applications [1], [2]. While these approaches can be effective in modeling changing appearance, they have the disadvantage that any tracking error will accumulate and propagate as the model is updated. Therefore, in any on-line adaptation method control of the adaptation is important. This is often done through setting a predefined confidence threshold for adapting to data [2] or introducing a *forgetting function* so newer data is used for adaptation [1]. While these methods have had some success they tend to be tuned for specific applications. The use of static appearance models avoids the problem of drift [3], [4], [5], however these models have difficulty in coping with changes in object appearance as tracking continues. There are two solutions to this problem: use sufficient training data to model appearance changes or construct an initial static model and adapt this online. Liu *et al.* [6] propose using a pre-trained

model of an object's appearance. This model is then updated online based on the appearance of the object. Multiple instance learning has been used to update pre-trained initial appearance models [7], [8]. As with purely adaptive models these approaches still find it difficult to control the adaptation and avoid drift. We take the approach of training a static appearance model offline. To generate sufficient training data we use a semi-supervised tracker to extract both positive (face) and negative (background) training examples from sequences containing only simple smooth motions and a single subject. This allows us to construct a model capable of robustly representing the full range of appearances of the object being tracked. As this appearance model is not updated online it is not affected by accumulated errors in the tracking.

Currently, one of the most effective methods for object appearance modeling in still images is dictionary learning (DL) or bag of visual words [9], which has shown state-of-the-art performance in many object recognition comparisons such as, the PASCAL Visual Object Class challenge [10] and the ImageCLEF Visual Concept Detection challenge [11]. DL methods have also been applied recently to the problem of tracking [6] where to model changes in appearance, a sparse coding histogram of the distribution of atoms for an image patch is adapted using the distribution of the target image patch. However, this method uses a learning rate parameter that is set *a priori*, this static learning rate does not account for changes in the tracking environment such as two similar objects occluding each other. Given that we can generate large amounts of training data we take the approach of training a static dictionary that is capable of representing all the variations in appearance found in the test set. Thus we avoid the problem of controlling adaptation in the appearance model. First we create a dictionary using K-means clustering, then use Soft Assignment (SA) methods to generate histograms or coefficient vectors. These vectors are then used to train a Support Vector Machine (SVM) classifier to discriminate face/head from background.

One drawback of using dictionary based methods in tracking applications is computational complexity. Liu *et al.* [6] propose a method known as K-selection to select a subset of atoms from the dictionary to represent an image patch. A gradient descent method is used to select the subset based on the location within the dictionary space. This method requires a search through all atoms in the dictionary to identify this subset for each feature vector at each time step in the tracking. For large dictionary sizes this may become prohibitively expensive. We also present a subset selection based method for improving the efficiency of histogram assignment using approximate Locality-constrained Soft Assignment (LcSA) [12]. The LcSA method has been shown to produce state-of-the-art results in the task of object recognition whilst giving a significant improvement in computational performance [12]. However, in contrast to K-selection, we employ a hierarchical dictionary structure to constrain our search space to a subset of dictionary codewords, based on Fast Hierarchical Nearest Neighbor Search (FHNNS) [12]. The LcSA method also improves the classification performance due to the sparseness of the histograms being more likely to render the classes linearly separable [13]. Therefore we can use the computationally efficient linear SVM [14], as opposed to more complex non-linear kernel based SVMs. The reason underlying this observation will also be studied by the sparsity index measures for different histogram generation methods (see Section VIII-B3).

In our application we are tracking multiple speakers moving around in a meeting room environment. This leads to the subject's occluding each other with possible loss of tracking or loss of the speaker's identity. Many tracking systems include subject identification to enable the tracking of multiple people [15], [16], [17], [18], however these approaches generally require a high resolution image of the face to perform well. Li *et al.* [19] proposed an on-line algorithm to adaptively model the identity of the subject, however they report difficulties in controlling the rate of adaptation. In our proposed tracking system we separate tracking from identification. We use a static appearance model for tracking thereby avoiding accumulation of errors, and we use an adaptive identity model for the more complex task of identity recognition. Due to the low resolution of our data, traditional face detection methods, such as those proposed by Viola-Jones [20], do not work, so instead we train a Gaussian Mixture Model (GMM) using data from around the subject's head. This data includes the context around the subject's head such as the clothes and background. We learn this identity on-line using Maximum a Posteriori (MAP) adaptation [21] to update the parameters of the GMM to account for new data. To overcome the problem of measurements being corrupted by data from another subject during occlusions, we use the distance between the subjects to control adaptation. If the subjects are far apart then more weight is given to the contribution of new data in updating the model parameters. As the subjects move closer together more weight is given to the prior distribution of the model parameters and ultimately adaptation is disabled. This is based on the fact that if the subjects are widely separated the data collected in the area of the subject's head will be more specific to that subject.

One current problem in most tracking applications is the initialization of the tracker. In the majority of cases the object to be tracked is simply manually selected in the initial frame of the sequence [22], [23], [24], [25], [26], [27]. In some cases a prior template or model is used to search in the initial frame for the object, for example a prior color template of a face is used by [28], [2], [29]. Alternatively a common face detection algorithm such as that proposed by Viola and Jones [20] can be used, as in the case of Naqvi *et al.* [30]. These methods require an exhaustive search of the initial frame and also if the number of objects to be tracked is not known *a priori* they can lead to false positive object detections. We propose a novel initialization method by using the audio azimuth angle for each speaker to constrain the search area for the visual face detector. We show that even a noisy audio tracker, discussed in Section VI-A1, combined with our general dictionary learning based face detector can be used for effective initial face localization.

The overall structure of this proposed system is outlined in Section II. In Section III we discuss the visual features used for dictionary construction. In Section IV we address the problem of appearance modeling using DL. Section V introduces methods such as LcSA and fast hierarchical clustering for improving the computational efficiency for dictionary based tracking. We describe how this DL based appearance modeling is integrated

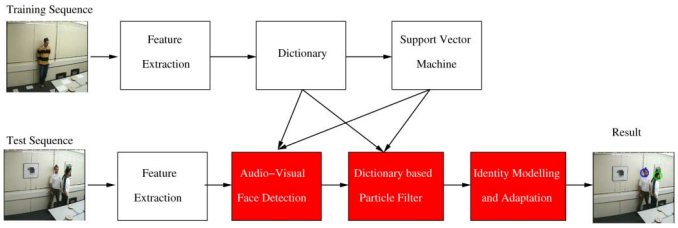

Fig. 1. Overall system to generate the 3-D head position, showing training and tracking or testing phases.

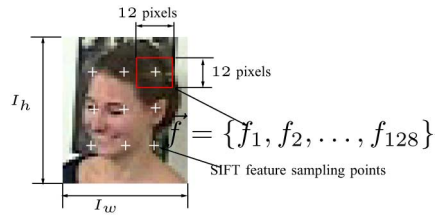

Fig. 2. Extraction of SIFT features from image patches. These are used in the training phase for dictionary construction and in the test phase for recognition.

into a PF framework in Section VI, including our proposed novel method of audio-visual face detection in Section VI-A. In Section VII we enable the tracking of multiple subjects using our proposed adaptive identity model. In Section VIII, we show the experiments conducted and the data used together with the results obtained. Conclusions are given in Section IX.

II. The Overall Structure of the Proposed System

In this section we present a system for tracking using a pre-trained dictionary and SVM classifier within the PF framework to provide robust and accurate three-dimensional tracking using multiple cameras. Fig. 1 shows the training and testing phases of our proposed tracking system, where the components of feature extraction, dictionary building and SVM classifier are standard, and the principal contributions of this paper are indicated in the shaded components. More specifically, in the testing phase we detect faces at the start of the sequence using our novel audio-visual face detector. Following this initialization we use our pre-trained dictionary and SVM classifier in the measurement step of the PF algorithm. Finally, we introduce a novel method of identity modeling and adaptation control to overcome occlusions.

III. Feature Extraction

As shown in Fig. 1, feature extraction is needed for both training and testing. A feature vector $\vec{f} = \{f_1, f_2, \ldots, f_M\}^T \in \mathbb{R}^M$ is a vector of transform coefficients for characterizing an image patch, where M is the feature dimension and T is a transpose. We extract two types of features from each image patch, the standard grey-scale SIFT and color histogram features of dimensions M_s and M_c respectively. SIFT features [31], which are histograms of gradient orientation, have been shown to be highly distinctive and also robust to affine image transformation [32]. Color histograms have many advantages in tracking applications being rotation and partially scale invariant, robust to partial occlusions, easy to calculate, and fairly robust to changes in illumination.

To calculate the SIFT feature vector, we densely sample the image patch with, typically, a horizontal step size of $I_w/3$ and a vertical step size of $I_h/3$, with the sampling points shown by the white crosses in Fig. 2, where I_w and I_h are the width and height of the image patch respectively. At each sampling point, we extract an image block of, typically, 12×12 pixels, from which we calculate the SIFT feature vector, $\vec{f} \in \mathbb{R}^{M_s}$. In practice, the adjacent image blocks may overlap with each other depending on the choice of I_w and I_h. We form the color feature vector $\vec{f} \in \mathbb{R}^{M_c}$ simply as a histogram of Hue values after transforming the image from the RGB color space to HSV space. The SIFT and color features are either used separately or concatenated to give a combined feature vector, $\vec{f} = \{f_1, f_2, \ldots, f_{M_{sc}}\}^T$, where $M_{sc} = M_s + M_c$. In our experiments while we test a number of different values for M_s and M_c, we typically choose $M_s = 128$, $M_c = 100$ for the majority of our experiments. As a result of the above calculation methods, for each image patch, we obtain nine SIFT, one color, and nine combined SIFT and color vectors. Note that, the combined feature vectors (for each image patch) are obtained by concatenating the same color vector with each of the nine SIFT vectors.

IV. Dictionary Learning Based Training Methods

A. Dictionary Construction

Based on the feature vectors extracted from image patches as in Section III, we form the training set as a matrix $\mathbf{F} = [\vec{f}_1, \ldots, \vec{f}_L] \in \mathbb{R}^{M \times \bar{L}}$ where $\bar{L}$ is the total number of feature vectors in the training set.[1] From $\mathbf{F}$, we can learn a dictionary $\mathbf{D} = [\vec{d}_1, \ldots, \vec{d}_U] \in \mathbb{R}^{M \times U}$, using e.g. the GMM algorithm, where $\vec{d}_u, u = 1, \ldots, U$, i.e. the so-called *visual codewords* (or *atoms*), and U is the total number of atoms in the dictionary. Such a dictionary provides a succinct representation of the feature vectors in $\mathbf{F}$.

In a GMM, each vector in the training set can be considered as a mixture of U Gaussian functions [33] with the following parameters to estimate, $\theta = (\theta_1, \ldots, \theta_U) = ((\omega_1, \vec{m}_1, \vec{\sigma}_1), \ldots, (\omega_U, \vec{m}_U, \vec{\sigma}_U))$, where ω_u, $u = 1, \ldots, U$, are the mixture component weights, $\vec{m}_u$ are the means and $\vec{\sigma}_u$ are vectors of the Gaussian component standard deviations. The density estimation problem can be addressed by optimizing the likelihood function $\Lambda(\mathcal{X}; \theta)$:

$$\Lambda(\mathcal{X}; \theta) = \prod_{l=1}^{\bar{L}} \sum_{u=1}^{U} \omega_u g(\vec{f}_l; \vec{m}_u, \vec{\sigma}_u), \tag{1}$$

where $g(\vec{f}_l; \vec{m}_u, \vec{\sigma}_u)$ is denoted as

$$g(\vec{f}_l; \vec{m}_u, \vec{\sigma}_u) = ([(2\pi)^M \cdot |\Sigma_u|]^{-\frac{1}{2}}) \times exp(-\frac{1}{2}(\vec{f}_l - \vec{m}_u)^T \Sigma_u^{-1} (\vec{f}_l - \vec{m}_u)), \tag{2}$$

[1] $\bar{L}$ is used to distinguish from L used later to denote the number of feature vectors extracted from each image patch.

where $|\Sigma_u|$ denotes the determinant of Σ_u and Σ_u is a diagonal co-variance matrix with diagonal entries σ_{ui}^2, $i = 1, \ldots, M$ being the elements of $\vec{\sigma}_u$.

Commonly the parameters of models such as GMMs are estimated through an iterative training algorithm such as Expectation-Maximization (EM). In practice, however, due to the simplifications detailed in the next section, we can directly allocate the parameters of the GMM from the parameters estimated by K-means clustering. Therefore, in our work, the means of the Gaussian mixtures are obtained from the dictionary codewords produced by the K-means clustering, $\vec{m}_u = \vec{d}_u$. The mixture weights, ω_u, are all set to the same value of $1/U$. The standard deviations are set to the same value such that $\sigma_{u1} = \sigma_{u2} = \ldots = \sigma_{uM} = \sigma$, where σ is estimated experimentally. As an example, the values of σ estimated for different dictionary sizes are described in Section VIII.

As there is little previous work using DL for this application we decided to test a number of different values for U in our experiments in Section VIII, to balance the number of visual words in the dictionary between being able to discriminate the object we wish to track and over-fitting on the training set. By over-fitting we mean that the model becomes over-specified on the training set and is unable to generalize to examples in the test set [34].

Our goal is, based on the dictionary $\mathbf{D}$, to create a compact representation of an image, or image patch, by using a coding coefficient vector (or histogram) $\vec{v} = \{v_1, \ldots, v_U\} \in \mathbb{R}^U$. The elements in $\vec{v}$ weight the contributions of each atom of $\mathbf{D}$ for coding the image, and are populated using a soft voting technique, as discussed next.

B. Histogram Generation Based on Soft Assignment (SA)

The simplest form of dictionary learning employs a vector quantization method known as Hard Assignment (HA). For each visual codeword $\vec{d}_u$ in the dictionary $\mathbf{D}$ the uth bin of the histogram $\vec{v}$ is assigned according to

$$v_u = \frac{1}{L} \sum_{l=1}^{L} \begin{cases} 1 & \text{if } \vec{d}_{\mathrm{u}} = \arg\min_{\vec{d} \in \mathbf{D}} (\mathbb{E}(\vec{d}, \vec{f}_l)) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $\mathbb{E}(\vec{d}, \vec{f}_l)$ is the Euclidean distance from the visual codeword $\vec{d}$ to the feature vector $\vec{f}_l$ and each bin is normalized by, L, the number of feature vectors extracted from an individual image patch. This is the simplest formulation for DL based classification methods. However, recent results in object recognition show that SA provides much better performance over HA [35], [36]. In SA, the expression of the membership probability, $\varrho_u(\vec{f}_l)$ of the component $\vec{m}_u$ being selected to represent $\vec{f}_l$ is given by:

$$\varrho_u(\vec{f}_l) = \frac{\omega_u g(\vec{f}_l; \vec{m}_u, \vec{\sigma}_u)}{\sum_{u'=1}^{U} \omega_{u'} g(\vec{f}_l; \vec{m}_{u'}, \vec{\sigma}_{u'})}. \tag{4}$$

The parameters of the model in equation (1) provide a vast number of degrees of freedom and therefore can be further reduced to $\theta = (\theta_1, \ldots, \theta_U) = ((\omega, \vec{m}_1, \vec{\sigma}), \ldots, (\omega, \vec{m}_U, \vec{\sigma}))$ by fixing all mixing weights $\omega_1 = \omega_2 = \ldots = \omega_U = \omega \neq 0$ to be equal and having a single $\vec{\sigma}$ parameter vector such that $\vec{\sigma}_1 = \vec{\sigma}_2 = \ldots = \vec{\sigma}_U = \vec{\sigma} \neq 0$. This yields the membership probabilities as follows:

$$\varrho_u(\vec{f}_l) = \frac{g(\vec{f}_l; \vec{m}_u, \vec{\sigma})}{\sum_{u'=1}^{U} g(\vec{f}_l; \vec{m}_{u'}, \vec{\sigma})} \tag{5}$$

Such a simplification renders a model that is more robust than the one given by equation (4) [36].

The uth bin of the histogram $\vec{v}$ representing an individual image patch is now calculated as

$$v_u = \frac{1}{L} \sum_{l=1}^{L} \varrho_u(\vec{f}_l). \tag{6}$$

The above SA formulation can be shown to be equivalent to the codeword uncertainty based SA method presented in [35].

C. Classifier Training

We have a number of histogram vectors with each being a sparse representation of an image patch in the training set, $\mathbf{V} = [\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_N]$, where N is the total number of image patches in the training set. These histograms which are produced by the processes described in Sections IV-B, V-A and V-B are then used as labelled training data to train an SVM classifier. Due to the sparsity of the histograms produced by these methods the two classes, head and background, are more likely to be linearly separable in a high dimensional space. This is confirmed in our experimental results in Section VIII where a binary linear SVM is used for classification.

V. Fast Algorithms for Histogram Generation From Dictionary Atoms

A. Approximate Locality-Constrained Soft Assignment (LcSA)

The SA technique described above does not explicitly minimize the error between $\vec{f}_l$ and its reconstructed version using the codewords from the dictionary. This can be addressed using Sparse Coding (SC) [37], [38] and Locality-constrained Linear Coding (LLC) [39], both aiming to optimize a cost function based on such an approximation error. However, the coding process in both SC and SA involves the whole set of the dictionary atoms, rendering potentially expensive computations. This can be a problem especially for a large size of dictionary, or for applications where computational load is a concern, as is our case. To address the limitations of SA, we adopt the notion of *locality* in coding, as used in LLC [39] and other recent methods [37], [38], [40], [41], [42], [43], by constraining codeword selection to the most relevant few.

We define the locality around $\vec{f}_l$, as the region of the dictionary space containing the c nearest codewords to $\vec{f}_l$, determined by the Euclidean distance. Specifically, we constrain SA to activate only c nearest codewords to the feature vectors as in [39], [43] when computing the membership probabilities. We refer to this variant of SA as approximate Locality-constrained SA (LcSA), i.e. finding c nearest codewords for reconstruction prior to the computation of assignments. Hence, LcSA obtains an *approximate* locality-constrained solution rather than a fully

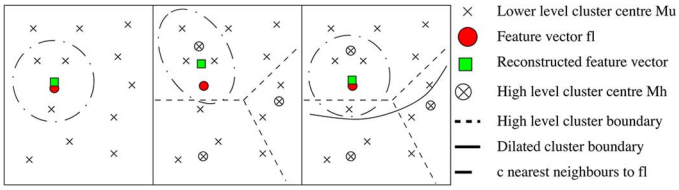

Fig. 3. Fast Hierarchical Nearest Neighbor Search. The left panel shows the reconstruction of a feature vector $\vec{f}_l$ using the c nearest dictionary codewords. The center panel shows the effect using hierarchical K-means to constrain the volume of the nearest neighbor search. The right panel shows that the reconstruction error can be reduced by dilating the boundaries of the higher level cluster centered on $\vec{m}_h$.

analytical one [12], and also achieves local smoothness and sparsity. To span local membership probabilities (as opposed to global ones as in equation (5)), one has to determine the c nearest neighbors (NNs) for every feature $\vec{f}_l$. Let us denote a subset of codewords as $\mathbf{D}_l^c = NN_{\mathbf{D}}(\vec{f}_l, c)$ where $NN_{\mathbf{D}}$ is a mapping of the c nearest codewords to the feature vector $\vec{f}_l$ from all the codewords in the dictionary $\mathbf{D}$. Limiting the membership probability in equation (5) to be based on only this subset of codewords $\mathbf{D}_l^c$ yields:

$$\varrho_u(\vec{f}_l) = \begin{cases} \frac{g(\vec{f}_l;\vec{m}_u,\vec{\sigma})}{\sum_{\vec{m}_{u'}\in\mathbf{D}_l^c} g(\vec{f}_l;\vec{m}_{u'},\vec{\sigma})} & \text{if } \vec{\mathrm{m}}_u \in \mathbf{D}_l^c \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

We can further consider the use of max-pooling for populating the histogram $\vec{v}$, representing an individual image patch, in the case of LcSA, so equation (6) is replaced with:

$$v_u = \max_l \varrho_u(\vec{f}_l), l = 1, \ldots, L \quad (8)$$

B. Fast Hierarchical Nearest Neighbor Search (FHNNS)

In Section V-A, the mapping function $NN_{\mathbf{D}}(\vec{f}_l, c)$ in LcSA is defined for the c NNs, with the search space still being the entire dictionary $\mathbf{D}$. The left panel of Fig. 3 shows the reconstruction of $\vec{f}_l$ as a linear combination of the c nearest codewords weighted by the coefficients from the histogram vector $\vec{v}$. This shows a small reconstruction error [12].

However, to improve computational efficiency in the NN search utilized by LcSA we constrain the search space of the mapping function. We employ a fast hierarchical NN search method by exploiting hierarchical K-means clustering [12]. First, we cluster the U codewords of the dictionary $\mathbf{D}$ into a dictionary, $\bar{\mathbf{D}}$, of H higher level codewords $\vec{m}_h, h = 1, \ldots, H$. We now define a subset of ρ_h codewords $\mathbf{D}_h^{\rho_h} = NN_{\mathbf{D}}(\vec{m}_h, \rho_h)$ which is composed of the ρ_h nearest lower level codewords $\vec{m}_u$ to the higher level codeword $\vec{m}_h$. If there is no overlap between the higher level clusters, that is, a lower level codeword can only belong to a single higher level cluster, then we have $U = \sum_{h=1}^{H} \rho_h$.

During histogram generation we define a new mapping function as $\vec{m}_h = NN_{\bar{\mathbf{D}}}(\vec{f}_l, 1)$ which gives $\vec{m}_h$, i.e. the closest high level codeword to the feature vector $\vec{f}_l$. We now define our subset of c codewords as, $c \leq \rho_h$, $\mathbf{D}_l^c = NN_{\mathbf{D}_h}(\vec{f}_l, c)$ where $\mathbf{D}_h$ is the set of ρ_h lower level codewords $\vec{m}_u$ within the cluster centered on the high level codeword $\vec{m}_h$. In practice, ρ_h, $h = 1, \ldots, H$, can all be chosen identical to ρ (found empirically in our experiments). The center panel of Fig. 3 shows the effect of using hierarchical K-means to constrain the volume of our NN search. It can be seen, however, that the reconstruction error can be larger due to the feature being unable to be represented by potentially more appropriate codewords across the boundaries of the selected higher level cluster.

To overcome this problem we propose dilating the boundaries of the higher level cluster centered on $\vec{m}_h$, used for the NN mapping, as shown in the right panel of Fig. 3. This relaxes the assumption that each lower level codeword can belong to only a single high level cluster, allowing overlap of the higher level clusters. The number of codewords in each high level cluster is now given by $\hat{\rho}$ where $\hat{\rho} > \rho$. The value of $\hat{\rho}$ is adjusted experimentally to achieve a balance between efficiency and accuracy. The reconstruction error in this case approaches that of the standard NN search [12], whilst still considerably reducing the search volume for the c nearest codewords. In practice this means that the histogram entry v_u for the codeword $\vec{m}_u$ will be zero if $\vec{m}_u$ is not within the set of c nearest codewords. A similar approach to fast NN search is employed by spill-trees [44].

C. Computational Efficiency Comparison

In this section we take a brief look at theoretical computational efficacy of HA, SA, LcSA and FHNNS, based on the results in [12]. HA can be easily described in terms of the NN search which scales linearly with the number of feature vectors to process, L, and the number of visual words to search through denoted as U. Thus, the complexity of HA amounts to $\mathcal{O}(L \times U)$. SA computes Gaussian-based distances from every feature vector to all available visual codewords. Next, it computes the sum of Gaussian distances. Lastly, it determines the ratio for every visual codeword to the total distance as in equation (5). Therefore, its complexity is $\mathcal{O}(L \times 3U) \approx \mathcal{O}(L \times U)$. LcSA is mainly limited by the NN search. This can be performed efficiently by the partial sort algorithm with a typical complexity $\mathcal{O}(L \times U \times log\ c)$, where c is a desired number of nearest codewords in searches. Summing distances and computing the ratio of Gaussians in equation (7) becomes an efficient operation with complexity $\mathcal{O}(L \times 2c)$. Therefore, the total assignment complexity is $\mathcal{O}(L \times U \times log\ c + L \times 2c) \approx \mathcal{O}(L \times U \times log\ c)$. Note, for sufficiently small $c \ll U$, LcSA becomes noticeably faster compared to SA. In our case $5 \leq c \leq 8$. The FHNNS further reduces the complexity of LcSA to approximately $\mathcal{O}(L \times \sqrt{U})$. This reduction in complexity from SA to LcSA is demonstrated in Section VIII-B3. Given the improved efficiency of LcSA, in the next section we describe how this is integrated into a PF framework for tracking.

VI. Modified PF Algorithm

The widely-used PF algorithm is modified here to incorporate the DL based histogram generation method described above. There are essentially four steps involved in a standard PF algorithm: initialization, propagation, measurement and re-sampling. Our new contributions are mainly in the initialization and measurement steps: an automatic initialization method of the visual tracker using audio information; and a novel method for computing the likelihood function in the measurement step based on LcSA (assisted by FHNNS) and SVM classification. The details of the proposed tracking algorithm are described below.

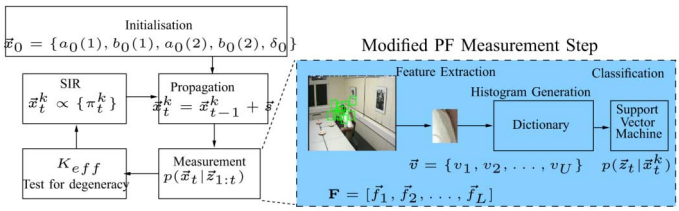

Fig. 4. Our modified DL based PF showing proposed changes to the measurement step in the shaded box.

We consider a dynamic system consisting of a hidden state sequence $\mathbf{X} = \{\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_T\}$, where T is the length of the sequence. This hidden sequence is the location through time t of the target speaker we wish to track where $t = 1, \ldots, T$. In our case each state is the position of a rectangular image patch centered on the subject's head, $\vec{x}_t^k = \{a_t^k(1), b_t^k(1), a_t^k(2), b_t^k(2), \delta_t^k\}$, where $a_t^k(1), b_t^k(1)$ and $a_t^k(2), b_t^k(2)$ are the coordinates of the top left and bottom right corners of the image patch for the kth particle respectively and δ_t^k is the velocity derived from the Euclidean distance from the center of the image patch defined by $\vec{x}_{t-1}^k$ to the center of $\vec{x}_t^k$. We also have a sequence of measurements $\mathbf{Z} = \{\vec{z}_0, \vec{z}_1, \ldots, \vec{z}_T\}$. In practice $\mathbf{X}$ is assumed to be a first order Markov process, so $\vec{x}_t$ depends solely on the previous state $\vec{x}_{t-1}$ and the current observation $\vec{z}_t$.

The task in tracking is to estimate the posterior probability distribution $p(\vec{x}_t|\vec{z}_{1:t})$. To this end, we generate a collection of K particles, $\mathbf{X}_t = \{\vec{x}_t^1, \ldots, \vec{x}_t^K\}$, each of which can be considered a hypothesis of the location of the target object, $\vec{x}_t$. Each particle also has an associated weight, giving a vector of particle weights $\vec{\pi}_t = \{\pi_t^1, \ldots, \pi_t^K\}$. The four steps of the PF algorithm are summarised in Algorithm 1 and Fig. 4. The details for each step are given in the following subsections.

Algorithm 1 Particle Filter for tracking a target state.

Input: $\mathbf{Z} = \{\vec{z}_0, \vec{z}_1, \ldots, \vec{z}_T\}$

Output: $\mathbf{X} = \{\vec{x}_0, \vec{x}_1, \ldots, \vec{x}_T\}$

$\vec{x}_0 = \{a_0(1), b_0(1), a_0(2), b_0(2), \delta_0\}$ % Initialization

while $t \leq T$ **do**

  for $k = 1$ to K **do**

    $\vec{x}_t^k = \vec{x}_{t-1}^k + \vec{s}$ % Propagate particles

    Calculate $p(\vec{z}_t|\vec{x}_t^k)$ using Algorithm 2

    $\pi_t^k = p(\vec{z}_t|\vec{x}_t^k) + p(\vec{x}_t^k|\vec{x}_{t-1}^k)$ % Measure particle fitness and update particle weights

  end for

  $\hat{\vec{x}}_t \approx \sum_{k=1}^K \pi_t^k \vec{x}_t^k$. % Estimate target position

  if $(K_{eff} > K_{effthresh})$ **then**

    Resample particles

  end if

end while

A. Audio-Assisted Automatic Initialization

In this section we address the problem of tracker initialization. This is very much an ongoing area of research in tracking and indeed most tracker systems rely on manual initialization. To initialize the tracker we must detect the initial head positions of the subjects. We did attempt to use one of the standard methods of face detection [20], however the results from this were disappointing on our dataset. This was possibly due to the small scale of the faces in our data and also the range of different head poses from each camera. Here we propose to exploit audio information and our general face model to initialize the tracker. Specifically, we use the direction of the speaker given by the audio tracker described in the following subsections to find automatically the initial head positions of the speakers in the room. This gives a collection of K particles at time $t = 0$ defined as $\{\vec{x}_0^1 \ldots \vec{x}_0^K\}$ all with the same image patch rectangle and the initial velocity of $\delta_0^k = 0$.

1) Audio Tracker: To find the approximate initial locations of the speakers, we employ the SAM-SPARE-MEAN method [45], which is an audio tracking algorithm developed for a smart meeting room environment. Other state-of-the-art audio tracking algorithms could also be used for this purpose, but are not considered here for two reasons. First, our focus is on the visual trackers, where audio tracker is used only for facilitating the initialization of the visual tracker. Second, our experiments in Section VIII indicate that even using a perfect tracker (i.e. annotated ground-truth) makes no difference in improving tracking performance of our proposed system.

The SAM-SPARE-MEAN algorithm is a two-step method. In the first step, the space around a circular microphone array is divided into a number of sectors, and the spectrum of the microphone signals is also discretized into a number of frequency bins. For each sector and frequency bin, the source activity (SAM), i.e. the posterior probability that at least one audio source, is estimated. In the second step, a parametric approach [46] is used for the localization of the sources (when detected as active in the first step), with the location parameters optimized with respect to a cost function such as SRP-PHAT [47].

2) Initialization of Visual Tracker: The azimuth angle produced by the audio tracker provides two very important pieces of prior information: the number of speakers and the general direction of each speaker, which are used here to constrain the number of trackers to be initialized and the area of the image to search for the face. To this end, we project a line in three dimensions from the center of the microphone array to a point (a, b, z), where a is equal to the distance from the center of the microphone array to the wall of the room in meters, denoted as R (which is 1.75 meters in our experiment, as shown on Fig. 7 in Section VIII), z can be estimated as the height of a human speaker, typically chosen as 1.80 meters in our experiment, and b is calculated as

$$b = \tan\left(\phi \times \frac{\pi}{180}\right) \cdot R \tag{9}$$

where ϕ is the azimuth angle (in degrees) of the speaker with respect to the circular microphone array shown in Fig. 7. The particle filter is initialized at intervals along this line to detect a face. The dispersal of particles gives a reasonable search area around this line. We then select the particle with the highest un-normalized weight as containing the subject's face. The sampling points and the initial face positions for the two multi-person sequences can be seen in Fig. 16 of Section VIII.

B. Particle Propagation

A particle filter assumes that $p(\vec{z}_t|\vec{x}_t)$ can be measured at a number of points, drawn from a proposal distribution $q(.)$, and so the distribution $p(\vec{x}_t|\vec{z}_{1:t})$ can be approximated by sampling these points. This sampling is performed in the propagation step of the PF as shown in Fig. 4 and in Algorithm 1. The particles are updated from time step $t-1$ to t, with the kth particle propagated according to the dynamic model

$$\vec{x}_t^k = \vec{x}_{t-1}^k + \vec{s}, \tag{10}$$

where $\vec{s}$, the transition noise, is a random variable with 2D Gaussian distribution with zero mean. Hence the particles are propagated based on their previous values and a certain amount of additive white Gaussian noise added to model the uncertainty in the motion involved.

C. Dictionary Learning Based Measurements

For each of these K particles produced by the propagation step, we generate a hypothesis of $p(\vec{z}_t|\vec{x}_t^k)$, for each of the current particle states $\{\vec{x}_t^1, \ldots, \vec{x}_t^K\}$. To generate this hypothesis we use the SA approach described in Section IV, and the FHNNS assisted LcSA approach from Section IV. So for each particle an image patch is extracted from the current frame using the coordinates $\{a_t^k(1), b_t^k(1), a_t^k(2), b_t^k(2)\}$. A set of features are extracted from the image patch as described in Section III. The pre-trained dictionary $\mathbf{D}$ is then used to produce a representation of the image patch according to equation (8), i.e. $\vec{v}^k = \{v_1^k, v_2^k, \ldots, v_U^k\} \in \mathbb{R}^U$, which is then classified by the pre-trained linear SVM in Section IV to obtain the likelihood of a particular particle's image patch containing a head as $p(\vec{z}_t|\vec{x}_t^k) = \mathbb{E}(\vec{v}^k, \vec{v}_{min}^k)$, where $\mathbb{E}(\vec{v}^k, \vec{v}_{min}^k)$ is the Euclidean distance from $\vec{v}^k$ to $\vec{v}_{min}^k$ the nearest point on the decision hyperplane of the pre-trained SVM. The weight of the kth particle at time t, π_t^k, is given by:

$$\pi_t^k = p(\vec{z}_t|\vec{x}_t^k) + p(\vec{x}_t^k|\vec{x}_{t-1}^k), \tag{11}$$

where $p(\vec{x}_t^k|\vec{x}_{t-1}^k)$ is a measure of the difference between δ_t^k and δ_{t-1}^k given by $1/|\delta_t^k - \delta_{t-1}^k|$. The particle weights are then normalized so that $\sum_{k=1}^K \pi_t^k = 1$.

Finally, the position of the speaker can be estimated as:

$$\hat{\vec{x}} \approx \sum_{k=1}^{K} \pi_t^k \vec{x}_t^k. \tag{12}$$

This gives us an updated estimate of the target position. Algorithm 2 summarizes the proposed DL based measurement step.

Algorithm 2 Dictionary learning measurement step.

Input: $\vec{z}_t$, K, L, U

Output: $p(\vec{z}_t|\vec{x}_t^k)$

for $k = 1$ to K **do**

Extract image patch at frame t according to $\{a_t^k(1), b_t^k(1), a_t^k(2), b_t^k(2)\}$;

Extract L features $\vec{f}_l, l = 1, \ldots, L$ from the image patch;

Create image patch representation $\vec{v} = \{v_1, v_2, \ldots v_U\}$,

where

$v_u = \max_l \rho_u(\vec{f}_l), l = 1, \ldots, L$;

Classify each image patch using SVM classifier to produce the likelihood $p(\vec{z}_t|\vec{x}_t^k)$.

end for

D. Degeneracy Testing and Particles Resampling

The method described above is known as *sequential importance sampling* (SIS). This sampling method leads to a problem known as degeneracy, where the weight π_t^k is concentrated in a single particle. This has the effect of dramatically degrading the approximation of the updated distribution. An effective measure of degeneracy is given by [48]:

$$K_{eff} = \frac{1}{\sum_{k=1}^K (\pi_t^k)^2}. \tag{13}$$

If all the weights of the sampled particles are equal, $\pi_t^k = \frac{1}{K}$, then $K_{eff} = K$. For re-sampling to take place a threshold is set on K_{eff} and if it rises above the threshold value the particles are re-sampled with probabilities proportional to their weights $\vec{x}_t^k \propto \{\pi_t^k\}, k = 1, \ldots, K$. This, known as Sampling Importance Re-sampling (SIR), eliminates particles with low weights and makes multiple copies of particles with high weights.

VII. Multi-Speaker Tracking via Adaptive Identity Modeling

The general pre-trained head model as described in Section VI is capable of differentiating head/face from background but not differentiating between faces. This leads to tracking errors when subjects approach and occlude each other. To overcome this problem we propose an adaptive model to recognize individual speakers, based on a GMM that is updated online using the MAP principle.

In a GMM the likelihood of the feature vector $\vec{f}_l, l = 1, \ldots, L$ is given by

$$p(\vec{f}_l) = \sum_{i=1}^{G} \omega_i g(\vec{f}_l; \vec{m}_i, \vec{\sigma}_i) \tag{14}$$

where $g(\vec{f}_l; \vec{m}_i, \vec{\sigma}_i)$ is a Gaussian distribution at $\vec{f}_l$, defined in equation (2), G is the number of Gaussian mixtures, ω_i, $\vec{m}_i$, and σ_i are the weights, means, and standard deviations of the

Gaussian mixture, respectively. Hence the GMM is fully parameterized by the set $\theta = \{W, m, \sigma\}$, where $W = \{\omega_i\}$, $m = \{\vec{m}_i\}$, and $\sigma = \{\vec{\sigma}_i\}$.

In online learning, as some data may not correspond to the correct label, prior knowledge is necessary to constrain the space of solutions for $\theta = \{W, m, \sigma\}$. This can be achieved using MAP adaptation, where prior knowledge is given by a prior distribution over θ, $p(\theta)$. Using the MAP principle we select θ such that it maximizes the *a posteriori* log likelihood,

$$\hat{\theta} = \arg\max_{\theta} p(\theta|\mathbf{F}) = \arg\max_{\theta} p(\mathbf{F}|\theta) + p(\theta), \tag{15}$$

where $\mathbf{F} = \{\vec{f}_1, \ldots, \vec{f}_{L\times N}\}$ is the set of data vectors. The contributions of the data likelihood, $p(\mathbf{F}|\theta)$, and the prior distribution, $p(\theta)$, can be balanced by introducing a weighting factor, α, in equation (15). So, we maximize $\alpha \cdot p(\mathbf{F}|\theta) + (1-\alpha)\cdot p(\theta)$, where α is a weighting factor on the prior parameters.

The parameters of the ith mixture of the GMM are adapted using the following set of update equations [21], [49]

$$\hat{\omega}_i = \alpha \cdot \omega_i^{pr} + (1-\alpha)\cdot \omega_i^{ml}, \tag{16}$$

$$\hat{\vec{m}}_i = \alpha \cdot \vec{m}_i^{pr} + (1-\alpha)\cdot \vec{m}_i^{ml}, \tag{17}$$

$$\hat{\vec{\sigma}}_i = \alpha \cdot (\vec{\sigma}_i^{pr} + Diag((\hat{\vec{m}}_i - \vec{m}_i^{pr})(\hat{\vec{m}}_i - \vec{m}_i^{pr})^T)) + (1-\alpha)\cdot(\vec{\sigma}_i^{ml} + Diag((\hat{\vec{m}}_i - \vec{m}_i^{ml})(\hat{\vec{m}}_i - \vec{m}_i^{ml})^T)), \tag{18}$$

where ω_i^{pr}, $\vec{m}_i^{pr}$ and $\vec{\sigma}_i^{pr}$ are the prior weight, mean and standard deviation, ω_i^{ml}, $\vec{m}_i^{ml}$, $\vec{\sigma}_i^{ml}$ are the parameters estimated by the maximum likelihood on the current data, and $\hat{\omega}_i$, $\hat{\vec{m}}_i$ and $\hat{\vec{\sigma}}_i$ are the updated parameters estimated by the MAP adaptation. The function $Diag(.)$ selects the diagonal elements of a matrix to form a single vector.

We use a GMM to model the head features of each subject. At the initial frame we extract a set of features from the automatically located head positions described in Section VI-A. A set of $L = 9$ feature vectors, described in Section III, are extracted from $N = 24$ image patches taken from the subject's face and body as shown in Fig. 5, these form a set of $L \times N = 216$ feature vectors. In the case of two subjects we have two sets of features $\mathbf{F}_0^0$ and $\mathbf{F}_0^1$, where the subscript 0 denotes the time index of the initial frame and the superscripts 0 and 1 represent the first and second subject respectively. These two feature sets are used to train two GMMs with parameters θ_0^0 and θ_0^1 respectively. At each subsequent frame a set of features are extracted from the subject's head position estimated by each of the trackers. So we have $\mathbf{F}_t^0$ and $\mathbf{F}_t^1$ at each time step t. We then estimate the likelihood of each of these sets of features given each of our GMMs, we select the subject's identity according to $\arg\max_j p(\mathbf{F}_t^j|\theta_t^0)$ for subject one and $\arg\max_j p(\mathbf{F}_t^j|\theta_t^1)$ for the second subject, where $j = 0, 1$. The data, $\mathbf{F}_t^0$ and $\mathbf{F}_t^1$, are also used to update the parameters of each identity model at each time step, according to equations (16), (17) and (18). Methods for controlling this adaptation are discussed in the following paragraph.

A key problem in MAP adaptation is the choice of $\alpha \in [0, 1]$, which controls the contribution of the prior parameters and the new parameters estimated from the current data. Instead of setting α at a fixed value as done usually, we adapt α according to the locations of the subjects being tracked as follows

$$\alpha_t = 1 - \frac{1}{I_w}\sqrt{\left(a_t^0(0) - a_t^0(0)\right)^2 + \left(b_t^1(0) - b_t^1(0)\right)^2} \tag{19}$$

where $(a_t^0(0), b_t^0(0))$ and $(a_t^1(0), b_t^1(0))$ are the head positions of the first and second subject at frame t respectively, I_w is the width of the image, and 0 in the bracket denotes the center of the image patch at the estimated head position. This formulation for α_t means that when the subjects are further apart we are more certain that the data collected relates only to that subject and so more weight is given to the new parameters estimated on the current data. The newly estimated parameters then become the prior parameters of the GMM.

Fig. 5. Feature extraction for subject identification. Note: The grids are deliberately shifted to avoid the colored balls on the subjects' heads which were used in annotation, instead of our tracking system.

Due to the low resolution of the video data used in our experiments, focusing solely on the subject's head was insufficient for identification. To overcome this we extract the features (as described in Section III) from a wider area around the head, including the context of the subject's clothing and background as shown in Fig. 5. Note that the balls in Fig. 5 were added to aid annotation but not used in our tracking systems. Due to the colored balls on the subject's heads we use the location of the grid on the lower half of the face instead of the center of the face. The initial prior distribution is trained using the initial head location which is found using the method described in Section VI-A2.

Essentially, when a subject is no longer recognized then the tracking in that camera does not contribute to the three-dimensional head position, however the position continues to be updated using the three-dimensional estimate of the head position produced by the other cameras. The identity of the subject continues to be tested using the three-dimensional position from the other trackers. Therefore, when the subjects move out of occlusion and the subject is again recognized, the tracker is turned on again.

VIII. Experiments and Results

To demonstrate the performance of our proposed approaches we conducted three sets of experiments for tracking in a real meeting room environment. Firstly, we evaluate the tracking performance of the DL based appearance modeling methods using HA, SA, and LcSA with FHNNS (based on equation (7)) respectively. The DL based histogram generation is compared with the baseline methods based on commonly used histograms of color or texture. The effect of different dictionary and feature sizes on the performance is also studied. The measurement step

of the PF is typically the distance between the histogram generated by image patch being tested and an exemplar histogram, typically the Bhattacharyya distance. The standard SA method proposed by van Gemert *et al.* [35] is used as a baseline in comparison with our proposed LcSA. We show that our system outperforms these baseline methods and is particularly robust to changing lighting conditions and large scale changes.

Secondly, we show the performance of our adaptive identity recognition method described in Section VII for tracking multiple subjects through occlusions. We show that having a pre-trained general face/head model combined with an adaptive identity model it is possible to track accurately multiple occluding subjects. We produce quantitative results for two and three subject tracking on the AV16.3 dataset [50]. In order to demonstrate that our proposed method can generalize to similar datasets we also present qualitative tracking results on the EPFL multi-camera pedestrian dataset [51] for three and four subjects and also a sequence from the CLEAR dataset [52] for five subjects. Finally, we show the performance of our proposed audio-visual face detection method described in Section VI-A2 for automatically and accurately detecting faces in the initial frame of the video sequence to be tracked. We demonstrate that our method outperforms the common baseline Viola-Jones face detection method [20] for tracker initialization.

A. Experimental Set-Up

The data used in our experiments consist of eleven annotated sequences from the AV16.3 dataset and also sequences from the EPFL pedestrian dataset and the CLEAR dataset. All the datasets feature multiple subjects recorded on multiple cameras in an indoor office or meeting room environment.

The AV16.3 dataset was recorded at the IDIAP research institute in 2004, in a smart meeting room environment using three calibrated cameras and two eight element omnidirectional circular microphone arrays. The data set was collected to specifically address the issues of large scale changes, natural illumination changes and partial and full occlusions. Within a single sequence the scale of the face/head may vary from approximately 50×70 pixels to 8×12 pixels, this can be seen from Fig. 6(a) to Fig. 6(c). The illumination changes within the meeting room can also be seen in Fig. 6. There are two main types of sequences in the AV16.3 dataset, meeting situations (two subjects seated at the table) and motion situations (subjects moving in the corner of the room). The position of the cameras was a compromise between these two situations [50], so camera one was situated to capture the faces of seated subjects and cameras two and three positioned to give a reasonable estimate of the 3-D position when the subjects are moving. We feel that the current challenges in tracking multiple people in an office environment are well represented in the AV16.3 dataset.

The data were annotated by using a simple color tracker that was manually corrected by a human observer. In a number of sequences colored balls were placed on the subjects' head to facilitate this process. However, we must stress that these colored balls played no part in our system, indeed we had to take particular pains to avoid them in the identity modeling experiments.

The layout of the smart meeting room with the locations of the three cameras and microphone array can be seen in Fig. 7. The

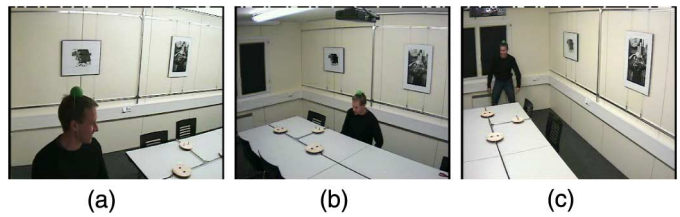

Fig. 6. Three images from sequence 11 from cameras 1, 2, and 3 respectively. (a) Camera 1. (b) Camera 2. (c) Camera 3.

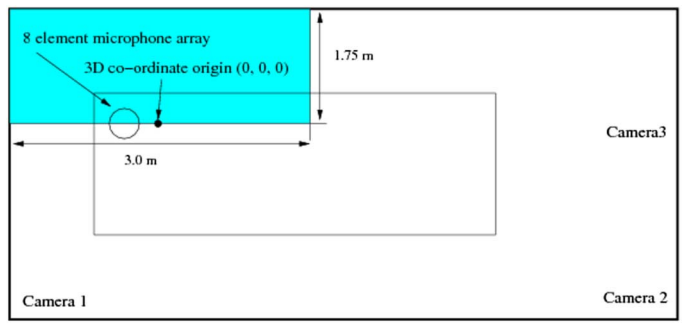

Fig. 7. Layout of room used for audio-visual recordings. The shaded area indicates the performance area for the subjects.

TABLE I
A SUMMARY OF THE DATA SEQUENCES USED FOR TRAINING AND TESTING. THE SEQUENCE NUMBERS CORRESPOND TO THE NUMBERING IN THE AV16.3 DATASET

Training Seq	Description
Seq 02	Single subject standing in predefined positions
Seq 03	Single subject standing in predefined positions
Seq 05	Single subject standing in predefined positions
Seq 06	Single subject standing in predefined positions
Testing Seq	Description
Seq 11	Single subject, moving randomly, abrupt head movements
Seq 12	Single subject, moving randomly, abrupt head movements
Seq 15	Single subject, traversing the room, smooth head movements
Seq 18	Two subjects, heads close with occlusions, sitting and standing
Seq 24	Two subjects, moving around the room with occlusions
Seq 40	Three subjects, two subjects seated while a subject moves behind them
Seq 45	Three subjects, all subjects moving with multiple occlusions

sequences feature subjects moving within the field of view of the three cameras and speaking continuously. The shaded area in Fig. 7 indicates the area within which the speakers move. The sequences vary in difficulty from the subject simply moving around a set of positions in the room with relatively constant direction and velocity, to multiple subjects moving freely around the room and making abrupt changes in direction. A summary of the data sequences used for training and testing can be found in Table I.

Each sequence is between 1000 and 3500 frames long with a frame rate of 25 frames per second and each video frame is a color image of 288×360 pixels. From these annotated sequences we selected four for training and five for testing. The variability of appearance in the training data was maximized by combining data from all three cameras to train a single model.

We take the approach of training a model of the subject to be tracked, in this case a person's head. An initial exemplar patch of the face is taken for each camera and the Battacharya distance is then calculated for each particle to determine its weight. This method is effective for tracking simple sequences and can be

TABLE II
RMSE IN METERS OBTAINED BY THE METHODS USING SIFT AND HUE HISTOGRAM AND BOTH HUE AND SIFT DICTIONARIES AND COMBINED HUE AND SIFT DICTIONARIES

Seq	Hue Hist	SIFT Hist	Hue Dict	SIFT Dict	Combined Hue/SIFT Dict
Seq 15	0.11	0.12	0.9	0.10	0.03
Seq 11	0.13	0.15	0.10	0.10	0.05
Seq 12	0.22	0.13	0.15	0.10	0.06

re-initialized by hand when it does fail. The background data was collected using the same frames as the head data. A single background image patch is extracted from each frame, this patch is selected as having the maximum Battacharya distance of all the particles. Using this method we can efficiently generate large amounts of varied training data. We apply the method to generate 37050 training examples, for both head and background. All tracking experiments were conducted with $K = 50$ particles. The tracking errors are measured using Root Mean Squared Error (RMSE), calculated as the Euclidean distance from the 3-D position estimated by the tracker to the 3-D annotated position of the subject's mouth.

B. Evaluation on Dictionary Learning Based Appearance Modeling

1) Comparison of Dictionary Versus Non-Dictionary Methods: For the dictionary based method, we first construct the dictionaries from the Hue and SIFT features, and then generate the histograms using these dictionaries (hence denoted as 'Hue Dict' and 'SIFT Dict') by the SA method described in Section IV-B. A combined color and SIFT dictionary was created and tested. The dictionary size was set to $U = 64$. The baseline methods that we compare are the non-dictionary methods i.e. using the Hue and SIFT histograms (hence 'Hue Hist' and 'SIFT Hist'). The results are shown in Table II. It can be seen that for all sequences the DL method based histograms provide better tracking performance as compared with using the Hue and SIFT feature vectors directly.

We also ran 50 random tests (randomly initialized dictionaries) to compare the three dictionary methods, i.e. 'Hue Dict', 'SIFT Dict' and 'Combined Hue and SIFT Dict' respectively, using a different random initialization for each one. Fig. 8 shows the tracking error for each frame of sequence 11 in the data, for a single instance of each tracking method. This plot shows the contribution of each of the modalities, it can clearly be seen that at the end of the sequence as the subject moves into an area with different illumination the errors in using the Hue based dictionary increase dramatically, while the 'Combined Hue and SIFT Dict' in which the contribution of the SIFT and Hue are given equal weighting shows a much smaller error. The SIFT based dictionary, on the other hand, performs poorly near the beginning of the sequence, which corresponds to the section where the subject sits down and moves into an area with more background clutter which can be seen in Fig. 6(b), however the 'Combined Hue and SIFT Dict' with a weighting of 0.5 from the Hue manages to overcome this problem. The 'Combined Hue and SIFT Dict' method performs robustly regardless of the subject's location in the room. This can further be confirmed by the error

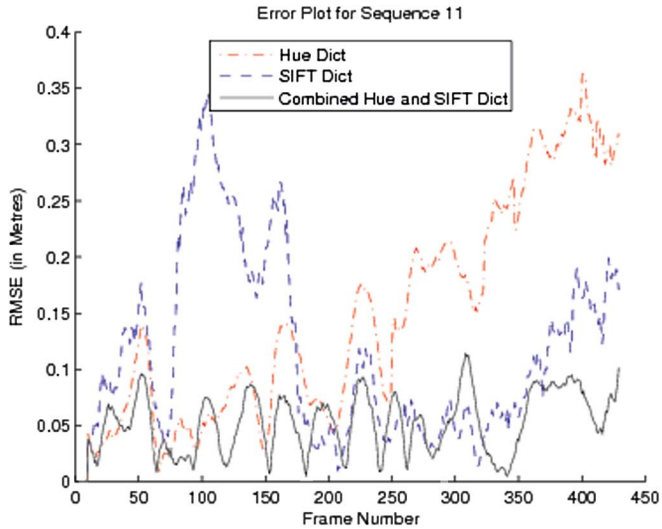

Fig. 8. The tracking results for sequence 11 of the AV16.3 database. This shows the performance of Hue and SIFT dictionaries and also the combined Hue and SIFT dictionary.

maps in Fig. 9 which shows the level of error for different locations within the room, the thicker the line the larger the error. The area of the plots in Fig. 9 corresponds to the shaded area in Fig. 7.

2) Comparison of Dictionary Size: We then evaluate the performance with respect to the dictionary size, U. Although in visual object recognition tasks larger dictionary sizes are commonly adopted we decided for practical purposes to limit the maximum size to 1024 atoms. We tested U from 32, 64, 128, 256, 512, to 1024. For each U, 50 random tests were performed, and the 'Combined Hue and SIFT Dict' based SA method was used. The average results of these random tests over the three single-subject test sequences (shown in Table II) can be seen in Fig. 10(a), where the error bars represent the standard deviation. It can be observed that for very small dictionary sizes, particularly for 'Hue Dict' and 'SIFT Dict', the results become very unstable due to the small number of dictionary atoms being unable to represent the face/head of the subject. However using the 'Combined Hue and SIFT Dict' the size of the error bars is much smaller, this may be due to the fact that a combined Hue and SIFT atom can represent more aspects of the data than a single Hue or SIFT atom. Interestingly, the best performance comes from the smaller dictionary sizes, this is probably due to the larger histogram size becoming overly sparse and degrading the head recognition performance.

We also compare the performance for different feature vector lengths. In this set of experiments we set the dictionary size to $U = 64$ and vary the length of the SIFT and color histogram feature vectors, M_s and M_c respectively. The values of M_s used are 32, 64, 128, 256 and 512 and the values of M_c used are 25, 50, 100, 200, 400. These two feature vectors are then concatenated as described in Section III to form a single combined feature vector of length $M_{sc} = M_s + M_c$. These different length features were then tested on sequence 11 of the AV16.3 dataset. The results of these experiments can be seen in Fig. 11, this plot shows that our selection of $M_s = 128$ and $M_c = 100$ is justified in terms of accuracy and computational feasibility.

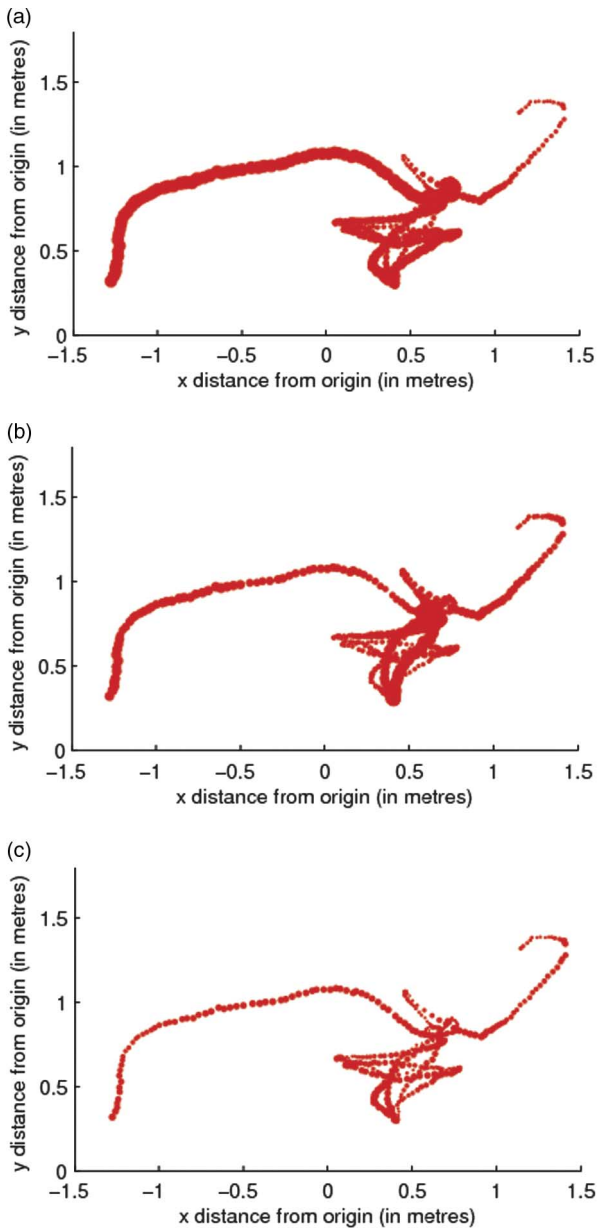

Fig. 9. Error maps for sequence 11 of the AV16.3 dataset. The area shown corresponds to the shaded area in Fig. 7. The thickness of the trajectory shows the average RMSE in that location, the thicker the trajectory of the plot the larger the RMSE. (a) Hue Dict Error Map. (b) SIFT Dict Error Map. (c) Combined SIFT and Hue Dict Error Map

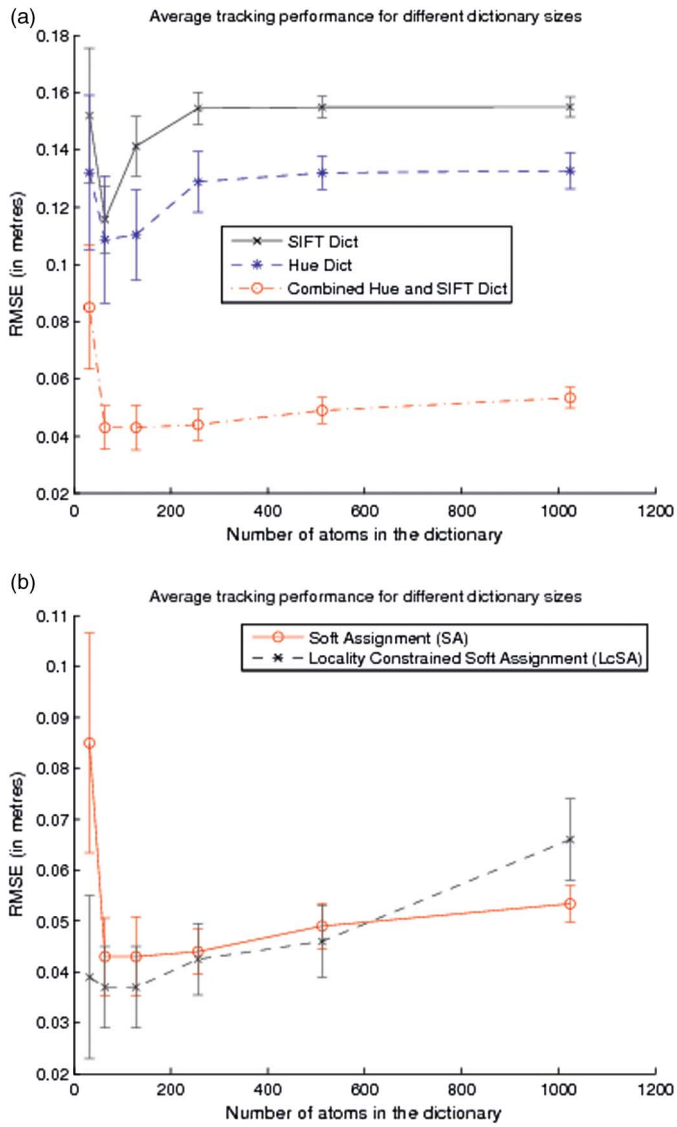

Fig. 10. Plot of RMSE averaged over 50 randomly initialized dictionaries and the three single subject test sequences for various dictionary sizes. The error bars show one standard deviation of the results from the 50 random tests. (a) Hue, SIFT and combined Hue and SIFT dictionaries using HA. (b) SA and LcSA dictionaries.

3) LcSA vs SA: One drawback of the SA method described in Section IV-B is that the contribution of all atoms in the dictionary is estimated to generate a histogram. In applications such as tracking this may cause problems with efficiency. In this set of experiments we test the performance of the LcSA method described in Section V for histogram generation to investigate whether we can achieve similar or better results by using a size reduced set of dictionary atoms for histogram generation.

The same set of training sequences used in Section VIII-B are used to create the set of training histograms using the method described in Section V-A. We use the 50 randomly initialized dictionaries created in the previous section with the combined Hue and SIFT features to create the LcSA histograms. The number of NNs, c, and the smoothing factor, σ, used to populate the histograms, according to equation (7), were determined using cross validation on the training sequences. The values of σ were set to 0.2 for $U = 32$ and 64, 0.19 for $U = 128$ and 256, and 0.18 for $U = 512$ and 1024. The number of NNs c was set to 5 for $U = 32$ and 64, 6 for $U = 128$, and 7 for $U = 256$, 512 and 1024. For dictionary sizes larger than $U = 256$ we implemented the fast NN search, FHNNS, described in Section V-B. The number of higher level codewords H was set to 128 for all dictionary size. The number of lower level codewords $\hat{p}$ was set to 128 for $U = 256$, and 256 for $U = 512$ and 1024.

In Fig. 10(b) we plot the average error for all 50 randomly initialized dictionaries over the three single subject test sequences for SA and LcSA for various dictionary sizes. It can clearly be seen that LcSA also gives a small improvement in the accuracy of the tracking. This could be explained by LcSA providing a

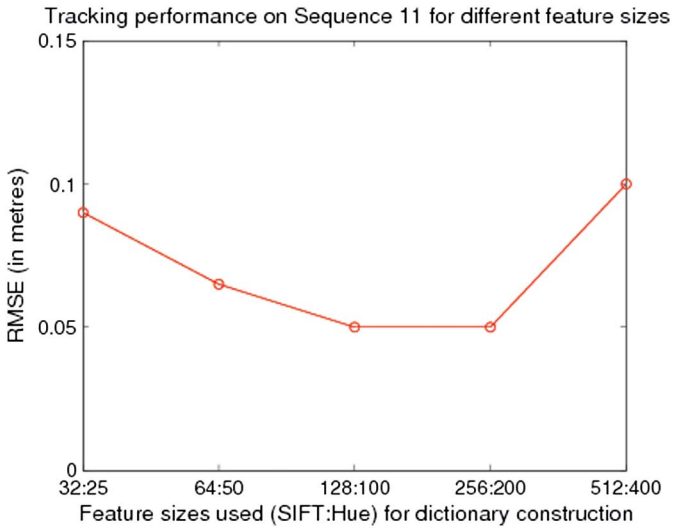

Fig. 11. The tracking result for different feature sizes for Sequence 11.

Fig. 12. Multiple person tracking using adaptive identity models. The images in the top row are from sequence 18, the middle row from sequence 24 and the bottom row from sequence 45.

more robust representation by selecting a smaller, but more relevant set of dictionary atoms, thus avoiding the noise introduced by less relevant atoms when using the entire dictionary. Again we can see that for small dictionary sizes the standard deviation of the results does increase however this effect is approximately the same as using LcSA. In order to measure the increase in tracking speed provided by the use of LcSA we measured the processing time for an individual frame with a single subject with a dictionary size of $U = 64$. Using the standard SA method the average frame processing time over 100 frames was 0.26 seconds, while using LcSA this processing time is reduced to 0.17 seconds. These experiments were performed using an Intel dual core 3 GHz desktop with 3.7 GB of memory. So we can see, our proposed system is capable of processing approximately 6 frames per second (despite the code not being optimized or fully parallelized).

Additionally, LcSA provides a more sparse representation than SA which may improve the classification of the linear SVM classifier. We use a common measure of sparsity given by $\tau_n = \frac{\|\vec{v}_n\|_1}{\|\vec{v}_n\|_2}$ where $\|\cdot\|_1$ and $\|\cdot\|_2$ are the ℓ_1 and ℓ_2 norms respectively and $\vec{v}_n$ is the nth image representation histogram. We then averaged this measure over all the histograms in the training set for different histogram generation methods. The results were LcSA (SIFT + Hue) $\tau = 3.9$, SA (SIFT + Hue) $\tau = 4.2$, SA (SIFT) $\tau = 4.6$ and SA (Hue) $\tau = 6.0$, where a smaller value of τ indicates a sparser vector.

C. Evaluation on Adaptive Identity Modeling on AV16.3 Dataset

Here we present results for tracking multiple speakers using the adaptive identity models described in Section VII. The experimental set up is similar to the single subject sequences, described in the previous section, with the main difference being that the dictionary based methods were automatically initialized using audio information extracted by the method described in Section VI-A2, as opposed to the manual initialization in Section VIII-B. The same images for dictionary building which were extracted from the four training sequences (see Table I) are used to train our GMM identity model using the EM algorithm [33]. After cross-validation on the training data we set $G = 10$, $N = 24$ in each frame (i.e. the 6×4 grid as shown in Fig. 5), and $L = 9$ for each image patch. This gives a total of $L \times N = 216$ feature vectors for training the GMM model for each frame. Both the SA and LcSA methods (based on the 'Combined Hue and SIFT Dict') were tested.

Sequences 18 and 24 (two subjects) and 40 and 45 (three subjects) of AV16.3 were used in this experiment, whose difficulty can be seen in Fig. 12. In sequence 18, the upper series of frames, the two subjects bring their heads together very slowly and then hold them very close for a number of seconds, this makes it difficult to use the subject's dynamics to overcome this type of occlusion. In our proposed system while we take into account the subject's motion we do not rely on it in order to be robust to occlusions. This allows us to track the subjects while the baseline method fails completely. The middle series of frames is from sequence 24 and shows the subject dressed in similar clothing, white t-shirts, despite this the identity of the subjects is preserved with our method through the occlusion. In theory if the subjects were dressed identically, with similar hair and skin color our identity modeling could fail, however we could not find such sequences in the AV16.3 dataset to test this, and this could be addressed in future work. The lower series of frames shows sequence 45 which features three speakers, all of whom are moving and occluding each other many times.

Table III shows the tracking results in RMSE for using the identity model, as compared with those without using the identity model. It can be seen that using the identity model, the tracking errors are considerably reduced. The error maps in Fig. 13 show that, as the subjects' heads come into close proximity the methods not using identity modeling fail whereas our proposed method continues to track both subjects. This can also be seen in Fig. 14 where both the SA and LcSA methods not employing identity modeling fail at the second occlusion. In addition, the tracking errors for the multiple person sequences are relatively higher as compared with those for single person

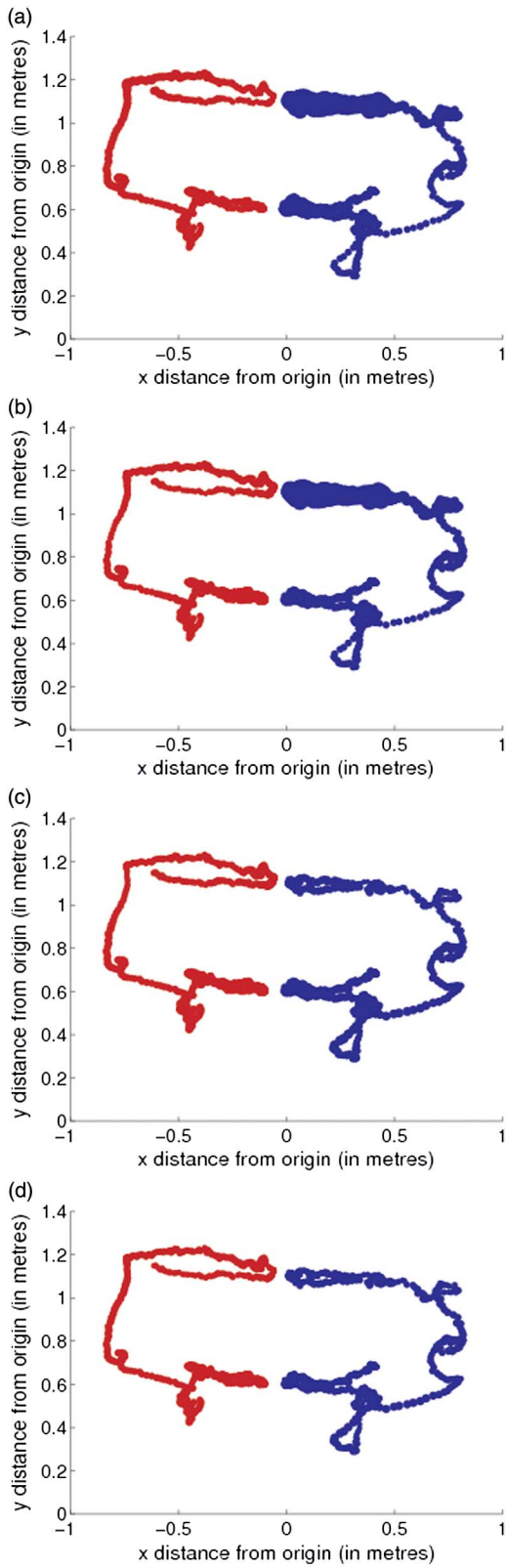

Fig. 13. Error maps for sequence 18 of the AV16.3 dataset. The area shown corresponds to the shaded area in Fig. 7. One subject is shown in red and the other in blue. The thickness of the trajectories depends on the average RMSE in that location, the thicker the trajectory the larger the RMSE. (a) SA Error Map (no identity). (b) LcSA Error Map (no identity). (c) SA Error Map (with identity). (d) LcSA Error Map (with identity).

sequences. The good performance of sequence 40 is given by the fact that two subjects are seated and stationary whilst

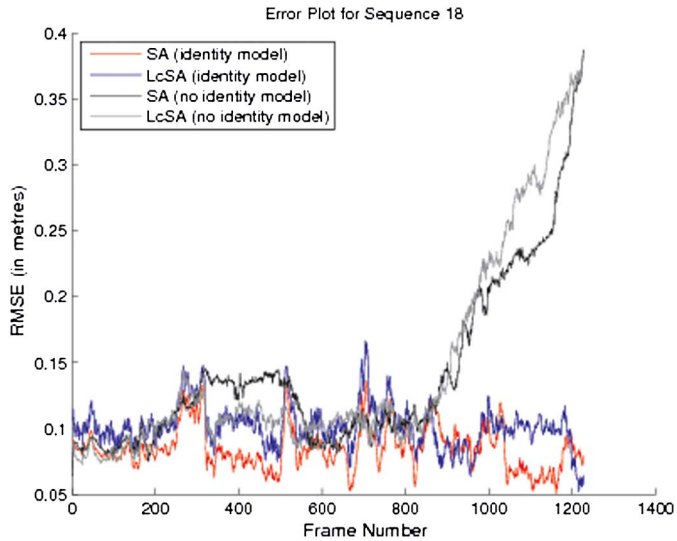

Fig. 14. The tracking results for sequence 18 of the AV16.3 database. This shows the performance of SA and LcSA method with and without identity modeling.

TABLE III
THE RESULTS FOR TRACKING SHOW THE RMSE IN METERS BETWEEN EACH MULTIPLE SUBJECT TEST SEQUENCE USING DL BASED SA AND LCSA METHODS, WITH AND WITHOUT IDENTITY MODELING

Sequence	SA	LcSA	SA (identity)	LcSA (identity)
Sequence 18	0.19	0.17	0.13	0.10
Sequence 24	0.11	0.10	0.09	0.09
Sequence 40	0.12	0.11	0.08	0.07
Sequence 45	0.35	0.34	0.15	0.14

the third subject moves behind them, there are no occlusions between the subjects.

It is worth noting that, after an extensive search of the current literature, we could only find two publications that reported visual tracking results for multiple subjects on the AV16.3 dataset. Pham *et al.* [53] exploit 3-D tracking with multiple cameras to overcome occlusion in sequence 24 and report an average accuracy of 0.06 meters, which is lower than our reported accuracy. However they do not report any results on the far more challenging sequence 18 which features slow moving subjects and partial occlusions. Khan *et al.* [54] report results for sequence 45 of the AV16.3 dataset, however their method involves tracking one subject in a multi-person environment and treating the other subjects as noise. So their results are not comparable with ours.

D. Evaluations of Multiple Subject Tracking on the EPFL and CLEAR Datasets

In order to show the ability of the system to generalize to data other than the AV16.3 dataset we selected sequences from two other indoor multiple subject tracking datasets, the EPFL multi-camera pedestrian dataset [51] and the CLEAR dataset [52]. In both cases the sequences feature multiple subjects moving in an indoor environment with multiple occlusions. We use the dictionary, SVM classifier and parameters trained/optimized on the AV16.3 dataset (dictionary size of $U = 64$ and $K = 50$ particles and the parameters for identity modeling are the same as those used in the previous section). Fig. 15 shows the results of tracking on the EPFL and CLEAR datasets. The first two

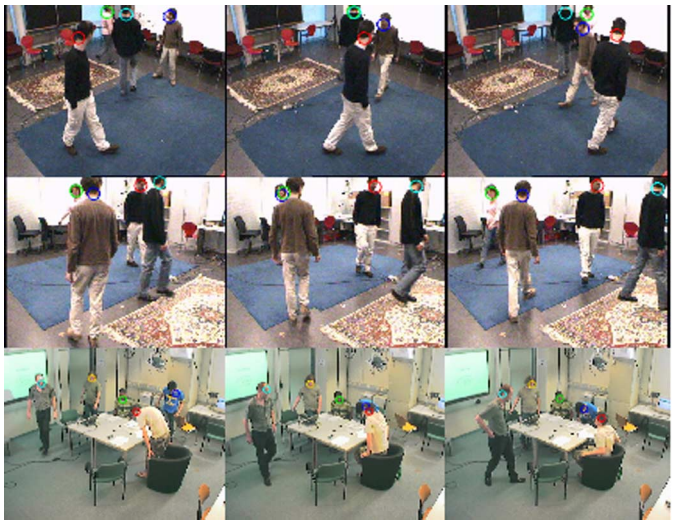

Fig. 15. Multiple person tracking using adaptive identity models. The images in the first two rows are from the EPFL dataset the bottom rows is a sequence from the CLEAR dataset.

rows of Fig. 15 show tracking from two cameras of the EPFL dataset. This shows the system accurately tracking four subjects against a complex background and through multiple occlusions. The final row shows the tracking of five subjects in a sequence from the CLEAR dataset, while the background is not as complex as the EPFL data there are still multiple occlusions of the subjects. The results also suggest that it is not necessary to repeat dictionary training for new data and camera configurations.

In both cases the tracking was manually initialized, for the EPFL data no separate audio signal was available and for the CLEAR dataset while audio data was provided none of the subjects starts speaking until they are seated and stationary. Also, no 3D information is available from the multiple cameras, so each of the tracking results are independent 2D results for each camera. This shows that strict calibration of the cameras is not necessary for our proposed tracking system to function.

E. Evaluations on Audio-Visual Tracker Initialization

To start tracking we must first locate the object or person we wish to track. Here we treat the initialization of the tracker as essentially a face detection and localization problem. To test our proposed method (described in Sections VI-A1 and VI-A2) we take the first frame in each sequence where the subjects' faces are visible and they are talking. In practice this does limit us to only detecting a face when the subject starts to talk, however this method of face detection demonstrates that the audio DOA can be useful in this task. The proposed system provides a flexible framework to incorporate other initialization methods, such as Viola-Jones, or other state-of-the-art face detection algorithms as a complementary way for reducing the possibility of the failure of initialization.

To provide a reasonable amount of data we annotated the initial face position on a total of 20 sequences from the AV16.3 dataset. These included 9 single and 11 multiple person sequences, giving a total of 84 faces. Each frame was annotated with a rectangle enclosing the subject's face. The audio was

TABLE IV
COMPARISON BETWEEN OUR PROPOSED AUDIO-VISUAL FACE DETECTION WITH THE VIOLA-JONES METHOD

Method	Precision ($Meas_p$)	Recall ($Meas_r$)
Viola-Jones	0.6	0.83
Proposed AV method (with estimated DOA)	0.97	0.97
Proposed AV method (with annotated DOA)	0.97	0.97

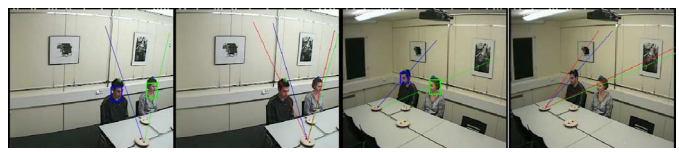

Fig. 16. Initialization of multi-person tracking sequences. Blue and green lines show the sampling line for the face detector the rectangle is the position of the particle with the maximum likelihood of a face. Also shown is the estimated DOA, green and blue lines, and the annotated DOA, red lines.

sampled at 16 kHz using a single 8 element circular microphone array with diameter 10 cm. The following parameters were fixed for all of our audio tracking experiments, time frame windows were 32 ms with an overlap of 16 ms. For the Fast Fourier Transform the number of samples was 512 and the number of histogram bins was 512. For the SRP-PHAT algorithm the number of sectors was fixed at 18 with each sector covering 20 degrees and the speed of sound was fixed at 320 m/s. Further details of the implementation can be found in [45]. We compare our method with one of the most common face detection algorithms proposed by Viola and Jones [20] implemented using the OpenCV computer vision library [55]. We used face images from the four training sequences (described in Table I) to set the thresholds for the Viola-Jones method.

To measure the performance of both methods we use precision and recall, where precision is given by $Meas_p = \frac{Num_c}{Num_r}$, where Num_c is the number of correct matches and Num_r is the number of potential faces identified by each method and recall is given by $Meas_r = \frac{Num_c}{Num_a}$ where Num_a is the number of faces in the frame. There are many ways to define what constitutes a correct face detection and this is often linked to the application and dataset being used. Rowley *et al.* [56] define a correct detection as the center of the detected face rectangle being less than four pixels from the center of the annotated rectangle and within 1.2 of the scale of the annotated rectangle. We follow a similar scheme in our experiments, however we adapt the measure to our particular data and application. As we are using the face detection algorithms to initialize a PF based tracker, in reality we can relax this measure as our pre-trained face model will converge to the face after a few iterations of the tracker. So we set the criteria for a face detection to be less than a Euclidean distance of 10 pixels from the center of the annotated face rectangle and a scale within 1.5 of the scale of the annotated rectangle.

Table IV shows the detection results for the 84 faces in the initial frames taken from 20 test sequences in the AV16.3 dataset. Fig. 16 shows the initial frames for each camera for the multiple subject sequences 18 and 24. The lines in the images show the tracks used by our proposed audio-visual face detection system

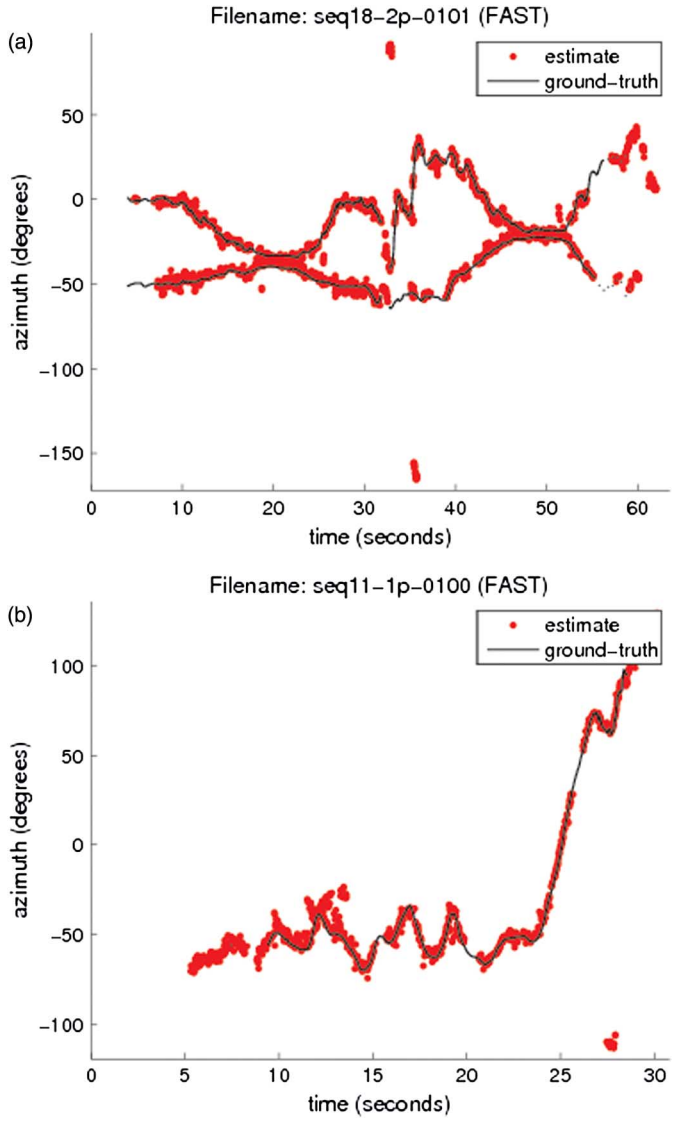

Fig. 17. Audio azimuth estimated by the audio tracking algorithm for sequences 18 and 11. (a) Sequence 18 audio azimuths. (b) Sequence 11 audio azimuths.

and the rectangle shows the estimated initial face location. It can be seen from the results that our proposed method performs significantly better than the baseline method. It seems most of the errors in the Viola-Jones approach come from an incorrect estimation of the number of speakers in the initial frame. The low precision of the Viola-Jones method is caused by the return of a number of false positives due to over estimating the number of faces in the frame. Audio information allows us to estimate *a priori* the number of speakers present, this information has the advantage of greatly reducing the rate of false positive detections.

It is worth noting that the audio signals used in our experiments were recorded in real room environments with the presence of room reverberations (($RT_{60} = 0.5$ seconds) and background noise. As a consequence, the accuracy of the DOA estimates are also degraded by such adverse acoustic effects. To see this, we show in Fig. 17 the output of the audio tracker for two annotated sequences (sequence 18 with two speakers and sequence 11 with a single speaker). It is interesting to note that, the estimation noise in DOAs has no adverse impact on the performance of our proposed visual tracking algorithm. To show this, we also performed the same experiment using the dataset annotations to provide a "perfect" estimate of the DOA, and the results are also included in Table IV. From this table, we can see that the detection result of the proposed algorithm (i.e. using the estimated DOAs for the initialization of the visual tracker) is identical to the result obtained by using the ground-truth DOAs for the initialization of the visual tracker. This implies that, even though using an up-to-date audio tracking algorithm may improve the audio tracking results, it does not improve the performance of our visual tracking algorithm. This is mainly because the estimated DOA is used in our algorithm to provide an estimate of the approximate speaker location and *a piori* the number of speakers. The accuracy of tracking is essentially achieved through the visual tracker. This can be further explained by Fig. 16, from which we can see that, even if there is noise in the estimated DOA (as in the left most subject), the result of the face detection offered by the audio tracker is sufficiently accurate for the initialization of visual tracker. For this reason, using other state-of-the-art audio trackers for the initialization of the proposed visual tracker is not considered here, we leave this to our future work.

IX. Conclusions

We have proposed a tracking system combining a DL approach for appearance modeling with a PF for dynamic modeling. We exploit the properties of DL to overcome the problems of recognition in low resolution images and under changing lighting conditions. This proposed method is shown to be more accurate than the baseline methods on the challenging AV16.3 dataset. We also demonstrate that the combination of Hue and SIFT features within a DL framework provides more robust tracking. The issue of the computational complexity of DL methods was addressed by proposing the use of LcSA for histogram generation. We show that using LcSA actually improved the performance of the DL based tracking system. A significant challenge in tracking is continuing tracking through occlusions. To overcome this we have introduced a method of identity modeling, this involves modeling the subject's appearance using a GMM and then adapting this model on-line using MAP adaptation controlled by the proximity of the other subject. We showed that this method combined with a DL based tracker can effectively track up to five subjects through occlusions whilst preserving their identity. We also demonstrated the ability of the system when trained using one dataset (AV16.3) to generalise to other datasets (EPFL and CLEAR) with no further training. Finally, we proposed an audio-visual face detection method for automatic tracker initialization. An audio tracker provides the DOA angle for each speaker and the number of speakers *a priori* thus greatly reducing the chances of a false positive face detection. We compared our proposed method to one of the standard methods for face detection [20], the results showed our method outperformed this baseline method on the challenging AV16.3 dataset.

REFERENCES

[1] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vision*, vol. 77, pp. 125–141, May 2008.

[2] K. Nummiaro, E. Koller-Meier, and L. J. V. Gool, "An adaptive color-based particle filter," *Image Vision Comput.*, vol. 21, no. 1, pp. 99–110, 2003.

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

[4] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2000, pp. 142–149.

[5] M. Ozuysal, V. Lepetit, F. Fleuret, and P. Fua, "Feature harvesting for tracking-by-detection.," in *Proc. Eur. Conf. Computer Vision*, 2006, pp. 592–605.

[6] B. Liu, J. Huang, L. Yang, and C. A. Kulikowski, "Robust tracking using local sparse appearance model and k-selection," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2011, pp. 1313–1320.

[7] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2009, pp. 983–990.

[8] J. Wang, X. Chen, and W. Gao, "Online selecting discriminative tracking features using particle filter," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2005, pp. 1037–1042.

[9] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Proc. IEEE Int. Conf. Computer Vision*, 2003, vol. 2, p. 1470.

[10] M. A. Tahir, J. Kittler, K. Mikolajczyk, F. Yan, K. Sande, and T. Gevers, "Visual category recognition using spectral regression and kernel discriminant analysis," in *Proc. Subspace Workshop in Conjunction with Int. Conf. Computer Vision*, 2009.

[11] M. A. Tahir, F. Yan, M. Barnard, M. Awais, K. Mikolajczyk, and J. Kittler, "The University of Surrey visual concept detection system at imageclef 2010: Working notes," in *Proc. Int. Conf. Pattern Recognition*, 2010.

[12] P. Koniusz, F. Yan, and K. Mikolajczyk, "Comparison of mid-level feature coding approaches and pooling strategies in visual concept detection," *Comput. Vision Image Understand.*, vol. 117, no. 5, pp. 479–492, 2013.

[13] J. Yang, K. Yu, Y. Gong, and T. S. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2009, pp. 1794–1801.

[14] N. Cristianini and J. Shawe-Taylor, *An introduction to Support Vector Machines*. Cambridge, U.K.: Cambridge Univ. Press, Mar. 2000.

[15] D. Gatica-Perez, G. Lathoud, J.-M. Odobez, and I. McCowan, "Audio-visual probabilistic tracking of multiple speakers in meetings," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 15, pp. 601–616, Feb. 2007.

[16] K. Bernardin and R. Stiefelhagen, "Audio-visual multi-person tracking and identification for smart environments," in *Proc. ACM Multimedia*, 2007, pp. 661–670.

[17] T. Chang, S. Gong, and E. Ong, "Tracking multiple people under occlusion using multiple cameras," in *Proc. British Machine Vision Conf.*, 2000.

[18] J. Czyz, B. Ristic, and B. M. Macq, "A particle filter for joint detection and tracking of color objects," *Image Vision Comput.*, vol. 25, no. 8, pp. 1271–1281, 2007.

[19] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade, "Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 10, pp. 1728–1740, 2008.

[20] P. A. Viola and M. J. Jones, "Robust real-time face detection," *Int. J. Comput. Vision*, vol. 57, no. 2, pp. 137–154, 2004.

[21] J. L. Gauvain and C. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Trans. Speech Audio Process.*, vol. 2, pp. 291–298, 1994.

[22] C. Wang and Z. Li, "A new face tracking algorithm based on local binary pattern and skin color information," in *Proc. Int. Symp. Computer Science and Computational Technology*, 2008, pp. 657–660.

[23] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Robust object tracking using joint color-texture histogram," *Int. J. Pattern Recognit. Artif. Intell.*, vol. 23, no. 7, pp. 1245–1263, 2009.

[24] J. Ye, Z. Liu, and J. Zhang, "A face tracking algorithm based on LBP histograms and particle filtering," in *Proc. Int. Conf. Natural Computation*, 2010, pp. 3550–3553.

[25] C. Shan, Y. Wei, T. Tan, and F. Ojardias, "Real time hand tracking by combining particle filtering and mean shift," in *Proc. IEEE Int. Conf. Automatic Face and Gesture Recognition*, 2004, pp. 669–674.

[26] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using sift features and mean shift," *Comput. Vision Image Understand.*, vol. 113, no. 3, pp. 345–352, 2009.

[27] Y. Tong, Y. Wang, Z. Zhu, and Q. Ji, "Robust facial feature tracking under varying face pose and facial expression," *Pattern Recognit.*, vol. 40, no. 11, pp. 3195–3208, Nov. 2007.

[28] D. N. Zotkin, R. Duraiswami, and L. S. Davis, "Joint audio-visual tracking using particle filters," *EURASIP J. Appl. Signal Process.*, vol. 2002, no. 1, pp. 1154–1164, Jan. 2002.

[29] Q. Cai and J. K. Aggarwal, "Tracking human motion using multiple cameras," in *Proc. Int. Conf. Pattern Recognition*, 1996, vol. 3, pp. 68–72.

[30] S. M. Naqvi, M. Yu, and J. A. Chambers, "A multimodal approach to blind source separation of moving sources," *IEEE J. Select. Topics Signal Process.*, pp. 895–910, 2010.

[31] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, pp. 91–110, 2004.

[32] K. Mikolajczyk and C. Schmid, "Comparison of affine-invariant local detectors and descriptors," in *Proc. Eur. Signal Processing Conf.*, 2004, pp. 1729–1732.

[33] J. Bilmes, A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models, Int. Comput. Sci. Inst., Berkeley, CA, USA, 1998, Tech. Rep.

[34] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. New York, NY, USA: Wiley, 1973.

[35] J. C. van Gemert, C. J. Veenman, A. W. M. Smeulders, and J. M. Geusebroek, "Visual word ambiguity," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 7, pp. 1271–1283, 2010.

[36] P. Koniusz and K. Mikolajczyk, "Soft assignment of visual words as linear coordinate coding and optimisation of its reconstruction error," in *Proc. Int. Conf. Image Processing*, 2011, pp. 2413–2416.

[37] H. Lee, A. Battle, R. Raina, and A. Y. Ng, "Efficient sparse coding algorithms," in *Proc. Conf. Neural Information Processing Systems*, 2007, pp. 801–808.

[38] J. Yang, K. Yu, Y. Gong, and T. S. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2009, pp. 1794–1801.

[39] J. Wang, J. Yang, K. Yu, F. Lu, T. Huang, and Y. Gong, "Locality-constrained linear coding for image classification," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2010, pp. 3360–3367.

[40] K. Yu, T. Zhang, and Y. Gong, "Nonlinear learning using local coordinate coding," in *Proc. Conf. Neural Information Processing Systems*, 2009, pp. 2223–2231.

[41] S. Gao, I. W. Tsang, L. Chia, and P. Zhao, "Local features are not lonely–Laplacian sparse coding for image classification," in *Proc. Conf. Computer Vision and Pattern Recognition*, 2010, pp. 3555–3561.

[42] X. Zhou, K. Yu, T. Zhang, and T. S. Huang, "Image classification using super-vector coding of local image descriptors," in *Proc. Eur. Conf. Computer Vision*, 2010, pp. 141–154.

[43] L. Lingqiao, L. Wang, and X. Liu, "In defence of soft-assignment coding," in *Proc. Int. Conf. Computer Vision*, 2011, pp. 2486–2493.

[44] T. Liu, A. W. Moore, A. Gray, and K. Yang, "An investigation of practical approximate nearest neighbor algorithms," in *Proc. Conf. Neural Information Processing Systems*, 2004, pp. 825–832.

[45] G. Lathoud and M. Magimai-Doss, "A sector-based, frequency domain approach to detection and localization of multiple speakers," in *Proc. Int. Conf. Acoustics, Speech and Signal Processing*, 2005, vol. 3, pp. 265–268.

[46] G. Lathoud, J. Bourgeois, and J. Freudenberger, "Sector-based detection for hands-free speech enhancement in cars," *EURASIP J. Appl. Signal Process.*, vol. 2006, pp. 169–184, 2006.

[47] J. DiBiase, H. Silverman, and M. Brandstein, "Robust localization in reverberant rooms," in *Microphone Arrays: Signal Processing Techniques and Applications*, M. Brandstein and D. B. Ward, Eds. New York, NY, USA: Springer Verlag, 2001.

[48] A. Kong, J. S. Liu, and W. H. Wong, "Sequential imputations and Bayesian missing data problems," *J. Amer. Statist. Assoc.*, vol. 89, no. 425, 1994.

[49] J. Mariéthoz and S. Bengio, "A comparative study of adaptation methods for speaker verification," in *Proc. Int. Conf. Spoken Language Processing*, 2002, pp. 581–584.

[50] G. Lathoud, J. M. Odobez, and D. Gatica-Perez, "AV16.3: an audio-visual corpus for speaker localization and tracking," in *Proc. Machine Learning for Multi-modal Interaction*, 2004, pp. 182–195.

[51] J. Berclaz, F. Fleuret, E. Turetken, and P. Fua, "Multiple object tracking using K-shortest paths optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 9, pp. 1806–1819, 2011.
[52] "The spring 2006clear evaluation and workshop," [Online]. Available: http://www.clearevaluation.org/.
[53] N. T. Pham, W. Huang, and S. H. Ong, "Probability hypothesis density approach for multi-camera multi-object tracking," in *Proc. Asian Conf. Computer Vision*, 2007, pp. 875–884.
[54] M. Khan, J. Ahmed, A. Ali, and A. Masood, "Robust edge-enhanced fragment based normalized correlation tracking in cluttered and occluded imagery," *Int. J. Adv. Sci. Technol.*, vol. 12, pp. 25–34, 2009.
[55] G. Bradski, "The OpenCV library," *Dr. Dobb's J. Softw. Tools*, 2000.
[56] H. Rowley, S. Baluja, and T. Kanade, "Neural network-based face detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 1, pp. 23–38, 1998.

Mark Barnard received a PhD from EPFL, Switzerland in 2005. Whilst completing his PhD he worked at The IDIAP Research Institute as a research assistant. His thesis was entitled Multimedia Event Modeling and Recognition. In 2006 he joined the Machine Vision Group at the University of Oulu where he was post-doctoral researcher for three years. He is currently a research fellow at the Centre for Vision, Speech and Signal processing at the University of Surrey. His current research interests include, audio-visual tracking, dictionary based image representation and audio head pose estimation.

Peter Koniusz (M'11) received the B.Sc. degree in Telecommunications, as well as in Architecture and Design of Microcontroller Systems from the Warsaw University of Technology, 2004. He completed his Ph.D. degree in Computer Vision at the University of Surrey (CVSSP) in 2013. He is currently a post-doctoral researcher at the team LEAR, INRIA, Rhône-Alpes. His interests include object category recognition, visual concept detection, action recognition, and multi-modal machine learning approaches.

Wenwu Wang (M'02–SM'11) received the B.Sc. degree in 1997, the M.E. degree in 2000, and the Ph.D. degree in 2002, all from Harbin Engineering University, Harbin, China. Since May 2007, he has been with the Centre for Vision Speech and Signal Processing, University of Surrey, Guildford, U.K., where he is currently a Senior Lecturer, and a Co-Director of the Machine Audition Lab. His current research interests include blind signal processing, sparse signal processing, audio-visual signal processing, machine learning and perception, machine audition (listening), and statistical anomaly detection. He has (co-)authored over 130 publications in these areas.

Josef Kittler (LM'12) is Professor of Machine Intelligence at the Centre for Vision, Speech and Signal Processing, University of Surrey. He conducts research in biometrics, video and image database retrieval, medical image analysis, and cognitive vision. He published a Prentice Hall textbook on Pattern Recognition: A Statistical Approach, as well as more than 170 journal papers. He serves on the Editorial Board of several scientific journals in Pattern Recognition and Computer Vision.

Syed Mohsen Naqvi (S'07–M'09) received his B.Eng. (first class) degree from IIEE/NED UET, Pakistan, in 2001 and his Ph.D. degree in Signal Processing from Loughborough University, Leicestershire, U.K., in 2009, where he is a Lecturer in the School of Electronic, Electrical, and Systems Engineering. He was a postdoctoral research associate on the EPSRC U.K. funded projects from 2009 to 2012. He contributed over 50 research outputs with main focus on multimodal (AV) speech processing and his research interests include nonlinear filtering, data fusion, and multi-target tracking, all for machine learning. He is a Member of the IEEE Signal Processing Society.

Jonathon Chambers (S'83–M'90–SM'98–F'11) received the Ph.D. degree in signal processing from the Imperial College of Science, Technology and Medicine, Imperial College London, London, U.K., in 1990. He currently heads the Advanced Signal Processing Group, School of Electronic, Electrical and Systems Engineering and serves as the Associate Dean (Research) LU in London. He has published more than 400 conference and journal articles, many of which are in IEEE journals. His research interests include adaptive and blind signal processing and their applications. Dr. Chambers is a Fellow of the Royal Academy of Engineering, U.K., and the Institution of Electrical Engineers. He has served on the IEEE Signal Processing Theory and Methods Technical Committee for six years. He has also served as an Associate Editor of the IEEE TRANSACTIONS ON SIGNAL PROCESSING and is currently a Senior Area Editor.