Distributed Face Recognition via Consensus on SE(3)

Roberto Tron, René Vidal

Distributed Face Recognition via Consensus on SE(3)

Roberto Tron and René Vidal

Center for Imaging Science, Johns Hopkins University, Baltimore MD 21218, USA

Abstract. We consider the problem of distributed face recognition in a calibrated camera sensor network. We assume that each camera is given a small and possibly different training set of face images taken under varying viewpoint, expression, and illumination conditions. Each camera can estimate the pose and identity of a new face using classical techniques such as Eigenfaces or Tensorfaces combined with a simple classifier. However, the pose estimates obtained by a single camera could be very poor, due to limited computational resources, impoverished training sets, etc., which could lead to poor recognition results. Our key contribution is to propose a distributed face recognition algorithm in which neighboring cameras share their individual estimates of the pose in order to achieve a "consensus" on the face pose. For this purpose, we use a provably convergent distributed consensus algorithm on $SE(3)$ that estimates the global Karcher mean of the face pose in a distributed fashion. Experiments on the Weizmann database show that our algorithm effectively improves the local pose estimates, and achieves the performance of centralized face recognition algorithms using only local processing.

1 Introduction

In the last few years, technological advances have made possible the construction of low-power devices that integrate on-board processing power with embedded cameras and wireless network interfaces. These devices can organize themselves into a "smart camera network" and constitute an attractive platform for deploying distributed algorithms. These new technologies enable us to re-think many computer vision applications, which traditionally employ centralized processing.

In face recognition, for example, most existing algorithms such as Eigenfaces [1, 2], Fisherfaces [3], ICA [4] and Tensorfaces [5], operate in a centralized fashion. In this paradigm, training is performed by a central unit, which learns a face model from a training set of face images. Images of a new face acquired from multiple cameras are also sent to the central unit, which performs recognition by comparing these images to the face model. However, notice that this approach has several drawbacks when implemented in a smart camera network. First, it is not fault tolerant, because a failure of the central processing unit implies a failure of the entire application. Second, it requires the transmission of huge amounts of raw data. Moreover, it is not scalable, because as the number of nodes in the network increases, so does the amount of processing by the central unit, possibly up to a point where it exceeds the available resources.

To the best of our knowledge, the development of distributed algorithms for face recognition has been fairly limited. Existing approaches [6] perform compression to reduce the amount of transmitted data, but the compressed images are still processed

by a central unit. In a fully distributed paradigm, each node could perform an initial processing of the images and transmit over the network only the distilled information relevant to the task at hand. Then, the nodes could collaborate in order to merge their local observations and come up with a global result, which is consistent across the entire network. This framework has several advantages over the centralized paradigm. For instance, when a single node fails the others can still collaborate among each other. Moreover, since each node performs its share of the processing, the aggregate resources available in the network grow with the number of nodes, making the solution scalable.

The fundamental question is how to develop distributed algorithms for face recognition? More specifically, what information should be extracted by the local nodes and shared among neighboring nodes? And more importantly, how should the local information be integrated within a theoretically sound framework so that a global objective can be minimized by the entire network?

In this paper, we choose the pose of the object as the local information to be extracted by the nodes and transmitted to neighboring nodes. This choice has several advantages. First, the pose of an object can be represented with only six parameters, which is obviously much less than the number of pixels in an image. Second, the pose of a face can be estimated from a single view using existing face recognition algorithms, e.g., view-based Eigenfaces [2]. Third, the pose of the face is a global quantity, which facilitates the definition of a global objective that all the nodes need to achieve, e.g., finding the average pose. The remaining question is then how to compute the average pose in a distributed fashion? *Consensus algorithms* (see e.g., [7, 8] and the references therein) provide a natural distributed estimation framework for aggregating local information over a network. In the classical *average consensus*, each node measures a scalar quantity, say temperature, and the average temperature over the entire network is obtained by iteratively updating the temperature reading at each node with the average temperature of its neighbors. Under mild network connectivity requirements, this iterative procedure is provably convergent to the global average. Unfortunately, existing consensus algorithms are not suited to computer vision problems. On the one hand, existing algorithms operate on low-dimensional measurements (e.g., temperature), which lie in a Euclidean space. On the other hand, images are high-dimensional, most of the quantities involved in the visual representation of a scene (pose, shape, etc.) are not directly measurable (i.e., they do not correspond to the images that the nodes see), and the images are related by parameters that belong to spaces with rich and complex non-Euclidean structures (e.g., Lie groups). Consequently, a principled treatment of such data requires the use of optimization on manifolds. While optimization on manifolds has been widely used in computer vision problems such as 3D reconstruction, 3D motion segmentation, shape clustering and manifold clustering, relatively less work has been done in extending consensus algorithms to non-Euclidean data [9, 10].

Paper contributions. This paper considers the problem of distributed face recognition in a camera sensor network. For the sake of simplicity, we will assume that the camera sensor network is accurately localized, i.e., each node knows its relative spatial transformation with respect to its neighbors. We will also assume that the face is visible to all the cameras and that it has been correctly detected in the images. Moreover, we will assume that a training set of face images taken under varying viewpoint, expression, and

illumination conditions is available. This training set can be given to each one of the camera nodes, or different subsets of the training set could be given to different nodes.

Our goal is to recognize the face by finding its pose in a distributed fashion. That is, we wish to combine the pose estimates coming from each camera into a global estimate which is consistent with the locations of the nodes across the network. In the solution proposed in this paper, we decouple the problem of estimating the individual poses from the problem of finding a global consistent estimate across the network. For the first part, we simply adapt appeareance based face recognition approaches, such as Eigenfaces and Tensorfaces, to estimate the pose of the face instead of its identity. For the second part, inspired by the work of [10], we propose to use distributed averaging algorithms (e.g., consensus) extended to data lying in a manifold (the space of rotations).

While our approach is specifically designed for recognizing faces, our distributed framework could be easily applied to other object recognition problems, as long as each camera node can provide an estimate of the object pose.

Paper outline. The remainder of the paper is organized as follows. §2 reviews two well known algorithms for face recognition, Eigenfaces and Tensorfaces, and shows how they can be adapted to estimate the pose of the face. §3 presents our framework for distributed pose averaging using consensus on $SE(3)$. §4 presents experiments on the Weizmann database, and §5 concludes the paper.

2 Review of Eigenfaces and Tensorfaces

Appearance based models have been among the simplest and most successful techniques for face recognition. In this section, we revisit the linear Eigenfaces algorithm [1] and its multilinear extension Tensorfaces [5], adapting them to the problem of recognizing the pose of the face rather than its identity.

2.1 Eigenfaces and View-based Eigenfaces

Consider an input image of dimension $h \times w$ as a vector in $\mathbb{R}^{hw}$. The main idea behind Eigenfaces [1] is that all face images lie in a common affine subspace of $\mathbb{R}^{hw}$, which can be learned from a training set of face images. Recognition of a new face image is performed by projecting it onto the "face subspace", and then comparing the coefficients of this projection with the coefficients of images projected from the training set using a nearest neighbor classifier.

In practice, the basic Eigenfaces model performs well only when the faces are viewed from similar pose, expression and illumination conditions. In order to accommodate significant variations of pose, the view-based Eigenfaces model in [2] assumes that all faces with the same pose, but different identity, illumination, expression, etc., lie in a common affine subspace of $\mathbb{R}^{hw}$. Recognition of new faces is then performed by projecting the new face onto multiple subspaces, one per pose. In this way one can find not only the identity of the face, but also its pose.

More specifically, the method proceeds as follows. Let $I = [I_{ip}]$ be a training set of face images, where the index $p = 1, \ldots, P$ identifies the face pose in the image while the index $i = 1, \ldots, N_p$ distinguishes images with the same pose p.

Training stage. The first step of the training stage is to decompose the training set I into P smaller training sets I_p containing the images corresponding to each pose. The next step is to center the data associated with each pose by removing the mean image $\bar{I}_p = \frac{1}{N_p} \sum_i I_{ip}$. Then, for each set I_p we compute the singular value decomposition (SVD) of the mean subtracted images $J_{ip} = I_{ip} - \bar{I}_p$ as

$$J_p = U_p \Sigma_p V_p^\top. \tag{2.1}$$

The matrices $U_p \in \mathbb{R}^{hw \times hw}$, $p = 1, \dots, P$, represent a basis for each face subspace. One can simplify this model by choosing the dimension of the subspaces, i.e., the number of Eigenfaces, to be $d_p \ll N_p \ll hw$. This can reduce the risk of overfitting the model to the training data and improve its generalization capabilities and recognition rates. For this purpose, we only need to consider the first d_p columns of U_p. This corresponds to finding the optimal rank-d_p approximation of the data matrix J_p with an error that has minimum Frobenious norm. In our experiments we considered only the case where all the subspaces have the same dimension, i.e., $d_p = d$.

Test stage. Given a test image I_t, its projection onto the p-th face subspace is

$$\tilde{I}_t = U_p U_p^\top (I_t - \bar{I}_p) + \bar{I}_p. \tag{2.2}$$

The reconstruction error is then defined as $e_p = \|I_t - \tilde{I}_t\|^2$, which is simply the distance of the test image from the p-th face subspace. The pose of the new face $\hat{p}$ can be obtained by finding the closest face subspace, i.e.,

$$\hat{p} = \operatorname*{argmin}_{p=1,\dots,P} \; e_p. \tag{2.3}$$

Given the pose $\hat{p}$, the identity of the new face can be found as the identity of the face in $I_{\hat{p}}$ which is closest to $\tilde{I}_t$.

2.2 Tensorfaces

Tensorfaces [5] is an extension of Eigenfaces in which all face images are assumed to lie in a multilinear variety, rather than an affine subspace of $\mathbb{R}^{hw}$. The multilinear variety is represented with a tensor, in the same way a subspace is represented with a matrix. Since the tensor has several dimensions, one can use one dimension to encode the different individuals, another to encode variations in pose, another for illumination, and so on. Therefore, Tensorfaces offers a natural way of modeling different kinds of variability, along with their interactions.

In the description below, we assume four modes of variation (identity, pose, illumination and expression) in addition to the dimensionality of the data (number of pixels). We denote the respective dimensions as N_{id}, N_{pose}, N_{illu}, N_{expr} and N_{pixels} or, for brevity, N_1, N_2, N_3, N_4, N_5.

Training stage. In the training phase we arrange all the images from the training set in a tensor $\mathcal{D} \in \mathbb{R}^{N_{id} \times N_{pose} \times N_{illu} \times N_{expr} \times N_{pixels}}$. We then perform the N-mode SVD, also known as Higher Order SVD (HOSVD) [11], which yields

$$\mathcal{D} = \mathcal{Z} \times_1 U_{pose} \times_2 U_{id} \times_3 U_{illu} \times_4 U_{expr} \times_5 U_{pixels}. \tag{2.4}$$

In this equation, $\times_k$ represents the tensor product along the k-th dimension. The matrices U_k are obtained by computing the SVD of the k-th matrix unfolding of the tensor $\mathcal{D}$, $D_{(k)} \in \mathbb{R}^{N_k \times N_{k+1} \ldots N_5 N_1 \ldots N_{k-1}}$, as $D_{(k)} = U_k \Sigma_k V_k^\top$. The rows of $D_{(k)}$ are indexed by the k-th dimension of $\mathcal{D}$, and its columns are indexed by all the remaining dimensions. The tensor $\mathcal{Z} \in \mathbb{R}^{N_{id} \times N_{pose} \times N_{illu} \times N_{expr} \times N_{pixels}}$ is called the core tensor, and is obtained by solving the linear system

$$\mathcal{Z} = \mathcal{D} \times_1 U_{pose}^\top \times_2 U_{id}^\top \times_3 U_{illu}^\top \times_4 U_{expr}^\top \times_5 U_{pixels}^\top. \tag{2.5}$$

For the details on how to perform these computations we refer the reader to [11].

In Eigenfaces the complexity of the model is controlled by the number of eigenfaces. In the case of Tensorfaces, we are dealing with high-order tensors and there is no unique way of extending this concept. A simple method is to find a rank-$(d_1, d_2, \ldots, d_n)$ approximation $\hat{D}$ of $\mathcal{D}$, with $d_k \ll N_k$, by restricting U_k to be of dimension $N_k \times d_k$ and $\mathcal{Z}$ to be of dimension $d_1 \times \cdots \times d_n$. For the sake of simplicity, in this paper we will define a number of tensorfaces d, set $d_k = \min(d, N_k)$, and compute $\hat{\mathcal{D}}$ using a rank-d_k SVD of each $D_{(k)}$. In our experiments we observed that performing more sofisticated choices of the ranks or performing iterative algorithms for factorizing $\mathcal{D}$ gave only marginal improvements on the approximation error and nearly identical recognition rates.

Testing stage. We can recognize the pose and identity of a new image by projecting it onto the "face variety" defined by the tensor $\hat{\mathcal{D}}$ and comparing the coefficients of this projection with those of projected images in the training set. In order to perform the projection, we define the tensor

$$\mathcal{B} = \mathcal{Z} \times_2 U_{id} \times_3 U_{illu} \times_4 U_{expr} \times_5 U_{pixels}. \tag{2.6}$$

Since the projection of one image from the training set with pose $pose$, identity id, illumination $illu$, and expression $expr$ is $\hat{\mathcal{D}}_{pose,id,illu,expr,(\cdot)}$, and $\hat{\mathcal{D}} = \mathcal{B} \times_1 U_{pose}$, the rows $c_{pose}^\top$ of U_{pose} are the "coefficients" of each pose in the "basis" $\mathcal{B}$. In fact, each $c_{pose}^\top$ generates, when multiplied by $\mathcal{B}$, a tensor containing all the images in the training set under the given pose.

Now, given a test image I_t, the computation of its coefficients requires nonlinear optimization, because we do not know the values for $id, illu, expr$. A sub-optimal solution can be obtained by extracting a subset of entries of $\mathcal{B}$ corresponding to fixed values for $id, illu, expr$, and flattering the resulting subtensor into a matrix $B_{id,illu,expr}$. For each $B_{id,illu,expr}$, we obtain a coefficient vector $\tilde{c}_{id,illu,expr}$ by finding the least square solutions to the linear systems

$$I_t = B_{id,illu,expr} \tilde{c}_{id,illu,expr}. \tag{2.7}$$

We then compute all the distances between each c_{pose} and all the $\tilde{c}_{id,illu,expr}$. The pose for which this distance is minimized is considered as the estimated pose. An analogous method can be used to find the indentity of the face, as shown in [5].

3 Face Recognition via Distributed Pose Estimation

In this section, we present our pose-based distributed face recognition algorithm. As before, we assume we are given a training set of face images from different individuals

and taken under different viewpoint, illumination, and expression conditions. Either the whole or small portions of this training set are given to each camera node. Small impoverished versions of the whole training set are more realistic, so as to improve the recognition speed of the nodes.

Given such impoverished estimates of pose, our goal is to obtain a *distributed method* for finding the pose in the training set that most closely resembles the one in the test image. For this purpose, we develop a distributed consensus algorithm that computes the average pose of the face from the local pose estimates. Given a global estimate of the face pose, we can estimate the identity as described in §2.

3.1 Distributed averaging consensus

Consider a sensor network modelled by a connected undirected graph $G = (V, E)$ where $V = \{v_1, \dots, v_N\}$ is the set of nodes and E is the set of edges. An edge $\{v_i, v_j\}$ belongs to E if and only if node v_i and node v_j are neighbors (i.e., they can communicate with each other). We define also the *adjacency matrix* A as

$$[A]_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \in E \\ 0 & \text{otherwise} \end{cases}. \tag{3.1}$$

Let $x_i \in \mathbb{R}$ be the state at node $i = 1, \dots, N$, which corresponds to a noisy estimate of a global quantity x. That is, $x_i = x + \epsilon_i$, where the ϵ_i's are zero-mean and i.i.d. The nodes are said to reach a *consensus* when $x_1 = \ldots = x_N = \alpha$.

A *consensus protocol* is a rule that the nodes use to evolve their states in order to reach consensus. An *average consensus algorithm* is a consensus algorithm which computes an average of the initial states. In this case the corresponding consensus protocol is given by

$$x_i(k+1) = x_i(k) + \sum_{j=1}^{N} a_{ij}(x_i - x_j), \qquad x_i(0) = x_i. \tag{3.2}$$

This protocol is obtained by performing gradient descent on the cost function

$$\varphi(x_1, x_2, \dots, x_N) = \frac{1}{2} \sum_{\{i,j\} \in E} a_{ij}(x_i - x_j)^2. \tag{3.3}$$

Since $a_{ij} = 0$ when $(i, j) \notin E$, the protocol in (3.2) requires only local computations. Furthermore, it is shown in [8] that this iterative protocol converges to the Euclidean average of the data. That is, for all $i = 1, \dots, N$, we have

$$\lim_{k \to \infty} x_i(k) = \alpha = \frac{1}{N} \sum_{i=1}^{N} x_i. \tag{3.4}$$

The average value α can also be interpreted geometrically as the point in $\mathbb{R}$ that minimizes the sum of squared Euclidean distances from the estimates, i.e.,

$$\alpha = \operatorname*{argmin}_x \sum_{i=1}^{N} (x - x_i)^2. \tag{3.5}$$

Notice also that extending the algorithm from $\mathbb{R}$ to $\mathbb{R}^n$ is straightforward, by applying the consensus algorithm in $\mathbb{R}$ to each coordinate of the state in $\mathbb{R}^n$.

3.2 Pose averaging in a camera sensor network via consensus

In a camera sensor network, the nodes of the graph $G = (V, E)$ correspond to the camera nodes, and the edges denote neighboring cameras. The state of the i-th camera node is the pose $(R_i, T_i) \in SE(3)$ of the face relative to the camera, where $R_i \in SO(3)$ is the face rotation and $T_i \in \mathbb{R}^3$ is the face translation. Our goal is to compute the global pose (R, T) of the face with respect to a fixed reference frame from the local estimates (R_i, T_i) in a distributed fashion. In order to extend the classical consensus algorithm from $\mathbb{R}$ to $SE(3)$, notice that $SE(3) = SO(3) \times \mathbb{R}^3$. Therefore, it is sufficient to develop a consensus algorithm for $SO(3) = \{R \in \mathbb{R}^{3\times 3} : R^\top R = I, \det(R) = 1\}$, because averaging consensus in $\mathbb{R}^3$ is straightforward. Since rotation matrices belong to a Lie group, the geometry of $SO(3)$ needs to be taken into account when generalizing the consensus algorithm from the Euclidean to the non-Euclidean case.

Centralized averaging of rotations. It can be shown that a proper distance in $SO(3)$ is the angle between the two rotations, which can be computed as

$$d(R_1, R_2) = \theta = \arccos\left(\frac{\mathrm{tr}(R_1^\top R_2) - 1}{2}\right), \quad R_1, R_2 \in SO(3). \tag{3.6}$$

This distance allows one to define a natural extension of the averaging operation from scalar values to rotations. Following the geometric interpretation of the average, one can define the Karcher mean [12] of N rotations $R_i \in SO(3)$ as

$$\bar{R} = \underset{R}{\mathrm{argmin}} \sum_{i=1}^{N} d^2(R, R_i). \tag{3.7}$$

This is the same notion as in the scalar case but, instead of using the Euclidean distance, it uses the geodesic distance in $SO(3)$. In general, the computation of $\bar{R}$ cannot be done in closed form as in the Euclidean case. We refer the reader to [13] for an iterative algorithm that computes $\bar{R}$ using gradient descent in $SO(3)$.

Distributed averaging of rotations in a camera sensor network. Let $h_0 = (S_0, t_0) \in SE(3)$ be an arbitrary reference frame. For the sake of simplicity, we choose $h_0 = (I, \mathbf{0})$. Let $h_i = (S_i, t_i),\ i = 1, \ldots, N$, be the poses of the nodes with respect to the reference frame. We will assume that the network is *localized*, i.e., each node $i = \{1, \ldots, N\}$ knows the pose of node j (where $\{i, j\} \in E$) with respect to its frame of reference. We use $h_{ij} = (S_{ij}, t_{ij}) \in SE(3)$, with $i = \{1, \ldots, N\}$, $\{i, j\} \in E$, to indicate the pose of node i in the reference system of node j. It can be easily verified that $h_{ij} = h_j h_i^{-1}$. This change of coordinates allows us to transform pose estimates from coordinate frame i to j as

$$(R_j, T_j) = (S_{ij} R_i, S_{ij} T_i + t_{ij}). \tag{3.8}$$

Recall from Eq. (3.3) that the classical consensus algorithm in (3.2) minimizes the sum of squared distances between the states of two neighboring nodes. Our objective is

essentially the same, except that (1) we need to use the geodesic distance instead of the Euclidean distance, and (2) we need to write our states $R_i \in SO(3)$ with respect to a common reference frame, before we can compute the distances. As a consequence, we propose to minimize the cost function

$$\varphi(R_1, \dots, R_N) = \frac{1}{2} \sum_{i,j=1}^{N} a_{ij} d^2(S_{i0} R_i, S_{j0} R_j) = \frac{1}{2} \sum_{i,j=1}^{N} a_{ij} d^2(S_{ij} R_i, R_j). \quad (3.9)$$

This cost function is analogous to the cost function in (3.3), except that the rotation at node i is brought into the coordinate system of node j as $S_{ij} R_i$.

We can minimize the cost function in (3.9) using a gradient descent algorithm in $SO(3)$. Since the covariant derivative (the equivalent of the gradient for a manifold) of $d^2(\cdot,\cdot)$ in $SO(3)$ is given by $\nabla_R d^2(R, Q) = -R \log(R^\top Q)$, we obtain

$$R_k^{(l+1)} = R_k^{(l)} \exp(\varepsilon \sum_{i=1}^{N} a_{ik} \log({R_k^{(l)}}^\top S_{ik} R_i^{(l)})). \quad (3.10)$$

Here $\log$ and $\exp$ are, respectively, the matrix logarithm and the matrix exponential. The following theorem states the convergence properties of (3.10).

Theorem 1. *The protocol in* (3.10) *converges to a local optimum of* φ *in* (3.9)*. When it converges to the global optimum, for all* $i, j = 1, \dots, N$, $\lim_{l\to\infty} S_{ij} R_i^{(l)\top} = R_j^* \in SO(3)$*. Moreover, when all the rotations* $S_{ij} R_j$*, are about a common axis,* R_j^* *is the Karcher mean of the* N *rotations in the reference frame of node* j*.*

Proof (Sketch only). Note that φ is the sum of non-negative factors and therefore $\varphi \geq 0$. Since $d(S_{ij} R_i, R_j)$ is zero iff $R_j = S_{ij} R_i$ and G is connected, the global optimum of φ is achieved when all the states converge to a single estimate of the face rotation consistent with the localization. If the rotations $S_{ij} R_i$ are all about a common axis, it can be shown that (3.10) preserves the Karcher mean iteration after iteration. Therefore, upon convergence, R_j^* must be equal to the Karcher mean of the initial data expressed in the reference frame of node j.

Experimentally, we have observed that the method always converges to the global minimum. Moreover, we have also observed experimentally that when the rotations are not about the same axis, the protocol converges to a first order approximation of the true mean (typically within an error of 0.1°).

In the case of switching networks (where the graph G – and therefore the matrix A – change over time) it has been shown in [8] that, under certain assumptions, the averaging consensus algorithm for Euclidean data (3.2) still converges. However, if this result holds also for the protocol extended to $SE(3)$ in (3.10) is an open question that will be left for future research.

4 Experiments

In this section we evaluate our algorithm on the Weizmann face image database. This database contains grayscale images of 27 different persons with variations of pose,

(a) Example of the different individuals. (b) Example of the different poses.

(c) Example of the different illuminations. (d) Example of the different expressions.

Fig. 1. Sample images from the subset of the Weizmann database used in the experiments.

illumination and expression. For our purposes we will use a subset of this dataset composed of 27 individuals, 5 poses ($0°$, $\pm 17°$ and $\pm 34°$), 4 illuminations (left, right, front, ambient) and 3 expressions. Expressions 2 and 3 are present for all the poses, but only under the first three illumination conditions. All the images can be considered to be already approximatively aligned. We resampled the images to 88×128 pixels and we did not perform any further processing. Figure 1 shows some examples of the variability present in the database.

Performance of single camera algorithms. In order to establish a baseline with which to compare the performance of our method, we first evaluated the single camera pose estimation algorithms described in §2. We divided the image dataset in two parts: the training set (users 1 to 21, all views, first three illumination conditions and one expression) and the test set (all the remaining images). The test set therefore contains images of the same people already present in the training set (but with different expressions and under different illumination conditions) and also of new people not present in the training set (users 22 to 27, all expressions and illumination conditions). The heterogeneous nature of the test set allowed us to evaluate how sensitive the different techniques for estimating the pose are to changes in the various factors: identity, illumination and expression. For this purpose we considered the following subsets of the test set:

1. *Same people, same illumination, new expression*: same people as in the training set under the same illuminations, but with different expressions;
2. *Same people, new illumination, same expression*: same people as in the training set and with the same expressions, but with different illuminations;
3. *New people, same illumination, same expressions*: people not present in the training set, same illuminations and expressions as in the training;
4. *New people, same illumination, new expressions*: people not present in the training set, new expressions, but same illuminations as in the training;
5. *New people, new illumination, same expressions*: people not present in the training set, same expressions as in the training, but different illuminations.
6. *New people, same illumination, all expressions*: the union of 3 and 4.

We first trained our models as described in §2. We then recorded the percentage of correctly recognized poses as a function of the dimension of the model for each one of the image sets listed above. We also added a test with the original training set in order to

verify that 100% of the faces are correctly classified when the same data is used for both training and testing and the full dimensionality of the model is used. The results of these experiments are shown in Figure 2 for both Eigenfaces and Tensorfaces.

Fig. 2. Pose recogntion rates as a function of the number of eigenfaces.

Notice that the pose recognition rate for all the methods on the original training set is perfect as expected. This shows that the model has been correctly learned. We can also see that all the methods are more sensitive to variations in illumination and identity than to variations in expression. Moreover, notice that in general Tensorfaces performs better than the simple Eigenfaces, but worse than the View-based Eigenfaces. This can be intuitively explained by considering that, on the one hand, Eigenfaces and Tensorfaces try to discover the pose of the face by using a nearest neighbour classifier, i.e., they compare the test image against single images of the training set. On the other hand, View-based Eigenfaces compare the test image against subspaces that are learned from multiple images, resulting in a more robust classifier. It would be of interest to evaluate a View-based version of Tensorfaces, but this issue is out of the scope of this work. In what follows, we will consider only Tensorfaces and View-based Eigenfaces. For the sake of brevity, from now on we will refer to the latter simply as Eigenfaces.

Performance of consensus with multiple cameras. In this experiment, two to five cameras are connected in a network with a ring topology. We used Eigenfaces and Tensorfaces to estimate the pose at each node and consensus to merge all the estimates. We want to evaluate the average rotation errors of the pose estimated by the camera network with respect to the ground truth pose.

In order to use the images from the Weizmann database to simulate the scenario where multiple cameras have images of the same face, we consider the variation in pose as the variation in viewpoint for the different cameras. For a given individual under given illumination and expression conditions, the different poses in the test images must be consistent with the localization of the network: for instance, if two of the cameras are separated by 17° degrees, then their two images will represent the corresponding variation in the pose by the same amount. In this experiment, given the number of cameras and their relative poses, we used as an input to the proposed method all the combinations of test images consistent with the localization. By averaging over all these results, we computed the mean and the variance of the rotation error.

Due to space limitations, we will report results from only two subsets of the test set, namely *New people, new illumination, same expressions* and *New people, same illumination, all expressions*. These results are, however, representative of results for the remaining subsets of the test set. The results of these experiments for both Eigenfaces and Tensorfaces are shown in Figure 3. The numbers in square brackets in the legend indicate the relative angle between two consecutive cameras. In order to check the validity of the distributed approach, we also repeated the experiment for *new people, same illumination, all expressions* using the centralized Karcher mean algorithm [13]. The results are in Figure 3 (e) and (f).

Notice that our distributed algorithm gives results perfectly equivalent to those of the centralized one. Notice also that, by averaging the local pose estimates, the mean and the variance of the error decrease as the number of cameras increases. The most appreciable improvements can be noticed in the variance of the error when the individual estimates are less reliable (e.g., in the case of Tensorfaces). Since the poses for testing are the same as those used for training, the average erros are in general relatively low (in the range $[0^\circ, 1^\circ]$ for Eigenfaces and $[2^\circ, 10^\circ]$ for Tensorfaces).

Performance of consensus as a function of the number of iterations. We will now look at how pose recognition and angular error vary as a function of the number of iterations of consensus. We fixed the dimensionality of the model to $d = 11$ for Eigenfaces and $d = 134$ for Tensorfaces. For each iteration, we collected the angular error and the recognition error at each node. The recognition error in this case is computed by first quantizing the current estimate of the rotation to one which is present in the training set and then comparing it with the ground truth. We took the average across all the trials and all the nodes. We considered only two subsets of the test set (*New people, same illumination, all expressions* and *New people, new illumination, same expressions*). As an additional experiment, we considered the case where each node has been trained with different individuals (about one fifth of the original training set). The results of these experiments are shown in Figure 4.

We can see that, in average, 40 iterations are sufficient for our method to converge. In general, the pose recognition rate increases and the angular error decreases as the number of iterations and the number of cameras increase. By comparing the final results of Figure 4 with Figure 2, we immediately see that multiple cameras give better pose recognition rates than a single camera. For instance, when the test set *new people, same illumination, all expressions* is used, the pose recognition rates increases from 93% (one camera) to 100% (five cameras) in the case of Eigenfaces and from 77% (one camera) to 100% (five cameras) in the case of Tensorfaces.

However, when the estimates from a single node are less reliable, e.g., Figure 4(b), the recognition error may decrease instead of increasing. In fact, if the majority of the pose estimates are wrong and not symmetric around the true pose, their average will give an incorrect classification, even if some of the initial estimates are correct. Also, notice from our experiments that sometimes using two cameras is better than using three or four. This, however, is probably an artifact of our limited dataset and it is probably due to the fact that configurations with more cameras have fewer sets of images on which we can collect the statistics. Nevertheless, networks with five cameras consistently gave the best recognition rates.

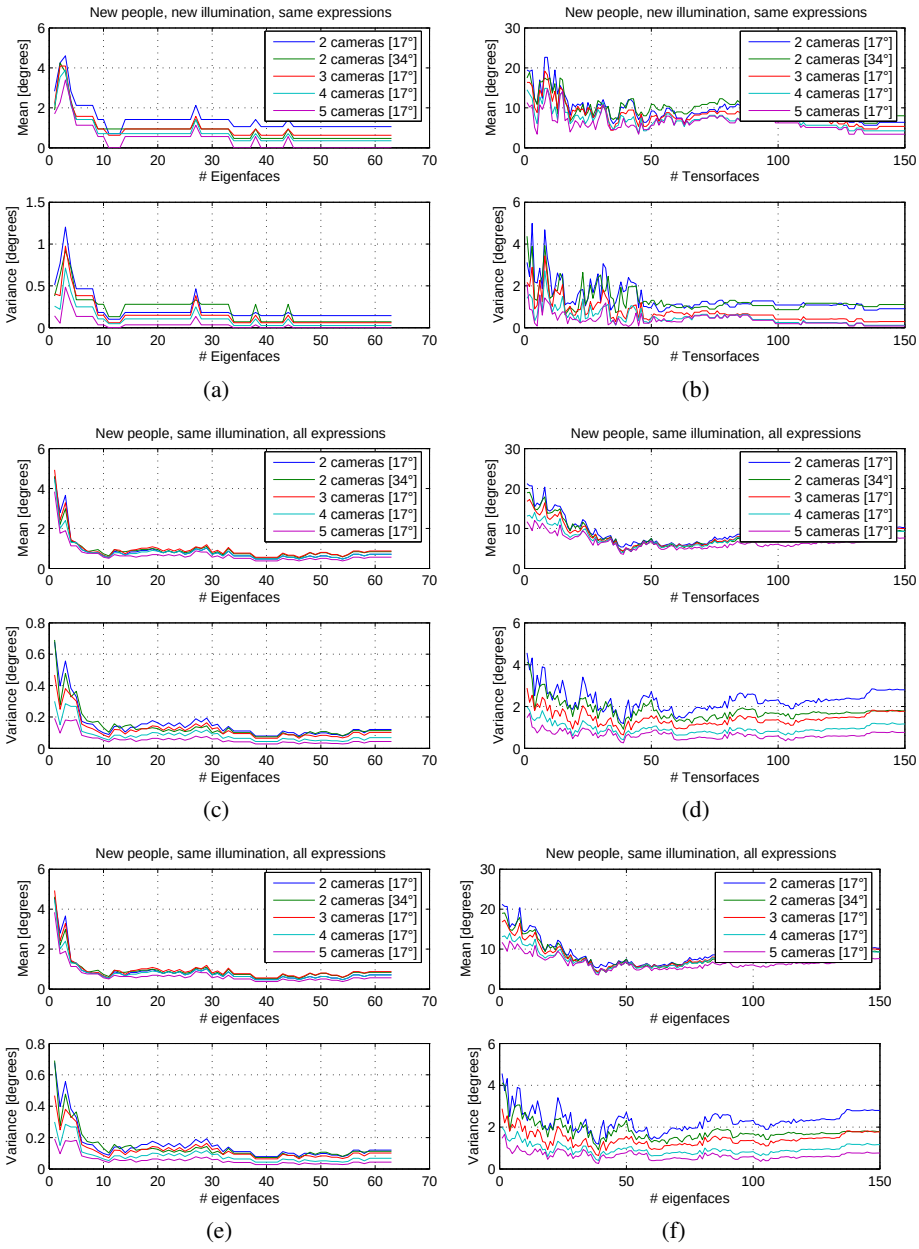

Fig. 3. Distributed algorithm: average and variance of the angle error between the aggregate estimation of the network and the ground truth as a function of the number of eigenfaces for the case of Eigenfaces ((a), (c), (e)) and Tensorfaces ((b), (d), (f)). Two subsets of the test set were considered: *New people, new illumination, same expressions* ((a) and (b)) and *New people, same illumination, all expressions* ((c) and (d)). For the latter we report also the results with the centalized algorithm ((e) and (f)): they match with the distributed case.

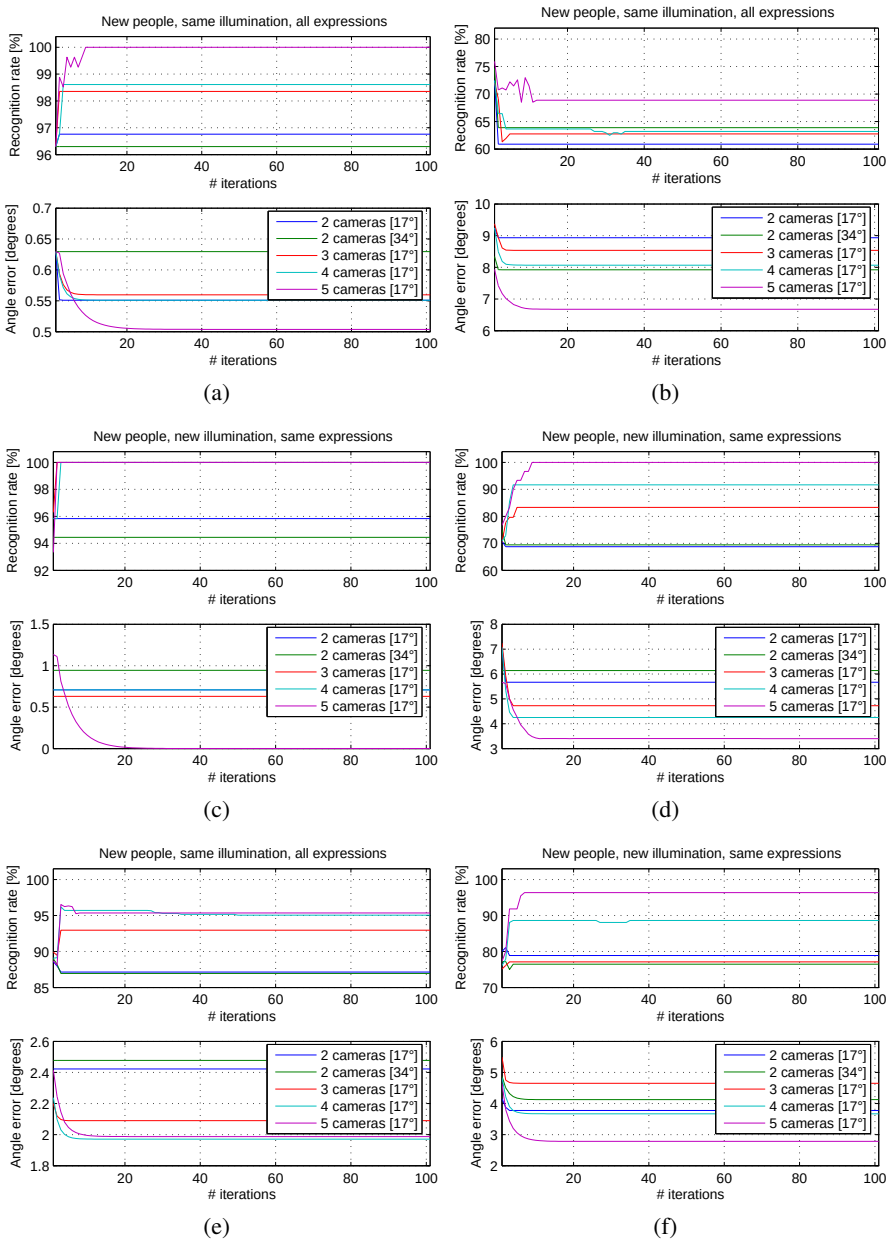

Fig. 4. Average pose recognition rates and angle error versus the number of iterations for consensus in the case of Eigenfaces ((a) and (b)), Tensorfaces ((c) and (d)) and Eigenfaces, different training set for each camera ((e) and (f)). Two subsets of the testing set were considered: *New people, same illumination, all expressions* ((a), (c), (e)) and *New people, new illumination, same expressions* ((b), (d), (f)).

5 Conclusions

We have presented an original distributed method for the estimation of the pose of a face in a smart camera network. Our method first uses appeareance-based methods to obtain pose estimates from the individual views and then uses an extension of the Euclidean consensus algorithm to $SE(3)$ for averaging these estimates. Our approach is totally distributed: each node extracts its estimate of the pose from the images using simple linear operations; then only the estimates of the pose are shared across the network using exclusively communications between neighbors. We evaluated the performance of our method using images from the Weizmann dataset under different settings (variations in illumination, expression, identity and number of nodes in the network).

At this point there are many possible areas on which our work could be extended. For instance, one can think of using different methods for estimating the pose of the face (e.g., by using Active Apperance Models). Also, it would be interesting to extend our approach to the case of uncalibrated cameras and switching networks. Moreover, it will be interesting to consider the problem of finding a distributed method for the general problem of classification.

References

1. Turk, M., Pentland, A.: Face recognition using eigenfaces. In: IEEE Conference on Computer Vision and Pattern Recognition. (1991) 586–591
2. Pentland, A., Moghaddam, B., Starner, T.: View-based and modular eigenspaces for face recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. (1994) 84–91
3. Belhumeur, P., Hespanha, J., Kriegman, D.: Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection. IEEE Transactions on Pattern Analysis and Machine Intelligence **16** (1997) 711–720
4. Bartlett, M., Movellan, J., Sejnowski, T.: Face recognition by independent component analysis. IEEE Transactions on Neural Networks **13** (2002) 1450–1464
5. Vasilescu, M., Terzopoulos, D.: Multilinear image analysis for facial recognition. In: International Conference on Pattern Recognition. (2002) 511–514
6. Serrano, N., Ortega, A., Wu, S., Okada, K., von der Malsburg, C.: Compression for distributed face recognition. In: Multimodal User Authentication Workshop. (2003)
7. DeGroot, M.: Reaching a consensus. Journal of the American Statistical Association **69** (1974) 118–121
8. Olfati-Saber, R., Fax, J., Murray, R.: Consensus and cooperation in networked multi-agent systems. Proceedings of the IEEE **95** (2007) 215–233
9. Olfati-Saber, R.: Swarms on sphere: A programmable swarm with synchronous behaviors like oscillator networks. In: IEEE Conference on Decision and Control. (2006) 5060–5066
10. Tron, R., Vidal, R., Terzis, A.: Distributed pose averaging in camera networks via consensus on SE(3). In: International Conference on Distributed Smart Cameras. (2008)
11. de Lathauwer, L., de Moor, B., Vandewalle, J.: A multilinear singular value decomposition. SIAM Journal of Matrix Analysis **21** (2000) 1253–1278
12. Karcher, H.: Riemannian center of mass and mollifier smoothing. Communications on Pure and Applied Mathematics **30** (1977) 509–541
13. Manton, J.: A globally convergent numerical algorithm for computing the centre of mass on compact Lie groups. In: International Conference on Automation, Robotics, Control and Vision. Volume 3. (2004) 2211–2216