Int J Comput Vis
DOI 10.1007/s11263-012-0558-z

Euler Principal Component Analysis

Stephan Liwicki · Georgios Tzimiropoulos · Stefanos Zafeiriou · Maja Pantic

Received: 5 December 2011 / Accepted: 8 August 2012

Abstract Principal Component Analysis (PCA) is perhaps the most prominent learning tool for dimensionality reduction in pattern recognition and computer vision. However, the ℓ_2-norm employed by standard PCA is not robust to outliers. In this paper, we propose a kernel PCA method for fast and robust PCA, which we call Euler-PCA (*e*-PCA). In particular, our algorithm utilizes a robust dissimilarity measure based on the Euler representation of complex numbers. We show that Euler-PCA retains PCA's desirable properties while suppressing outliers. Moreover, we formulate Euler-PCA in an incremental learning framework which allows for efficient computation. In our experiments we apply Euler-PCA to three different computer vision applications for which our method performs comparably with other state-of-the-art approaches.

Electronic supplementary material The online version of this article (doi:10.1007/s11263-012-0558-z) contains supplementary material, which is available to authorized users.

S. Liwicki (✉) · G. Tzimiropoulos · S. Zafeiriou · M. Pantic
Department of Computing, Imperial College London,
180 Queen's Gate, London SW7 2AZ, UK
e-mail: sl609@imperial.ac.uk

S. Zafeiriou
e-mail: s.zafeiriou@imperial.ac.uk

M. Pantic
e-mail: m.pantic@imperial.ac.uk

G. Tzimiropoulos
School of Computer Science, University of Lincoln, Brayford Pool, Lincoln, LN6 7TS, UK
e-mail: gtzimiropoulos@lincoln.ac.uk

M. Pantic
Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, Enschede, The Netherlands

Keywords Euler PCA · Robust subspace · Online learning · Tracking · Background modeling

1 Introduction

In pattern recognition, Principal Component Analysis (PCA) is perhaps the most classical tool for dimensionality reduction and feature extraction. It is widely utilized in a great variety of disciplines, including agriculture, biology and economics (Jolliffe 2002). Researchers in computer vision employ PCA for face recognition (Turk and Pentland 1991), object tracking (Ross et al. 2008), background modeling (Li 2004) and many other applications (Jolliffe 2002). It has been primarily used for efficient dimensionality reduction such that most of the variance of the original high dimensional data is preserved.

Given a population of n samples $\mathbf{X} = [\mathbf{x}_1 \cdots \mathbf{x}_n] \in \mathbb{R}^{p \times n}$ in a p-dimensional vector space,[1] standard PCA finds a set of $m \leq p$ (usually, $m \ll p$) orthonormal basis functions $\mathbf{B} = [\mathbf{b}_1 \cdots \mathbf{b}_m] \in \mathbb{R}^{p \times m}$ by minimizing the error function φ with respect to $\mathbf{B}$

$$\mathbf{B} = \arg\min_{\check{\mathbf{B}}} \varphi(\check{\mathbf{B}}) = \arg\min_{\check{\mathbf{B}}} \left\| \mathbf{X} - \check{\mathbf{B}}\check{\mathbf{B}}^T \mathbf{X} \right\|_F^2, \tag{1}$$

where $\|.\|_F$ denotes the Frobenius norm. It can be shown (Jolliffe 2002) that the solution is given by the eigenvectors corresponding to the m largest eigenvalues obtained from the eigendecomposition of the covariance matrix $\mathbf{S} = \mathbf{X}\mathbf{X}^T$ (or the Singular Value Decomposition (SVD) of $\mathbf{X}$).

The ℓ_2-norm in (1) is optimal for the case of independent and identically distributed (i.i.d.) Gaussian noise but not robust to outliers (de la Torre and Black 2003; He et al. 2011;

[1] Without loss of generality we assume zero mean.

Published online: 05 September 2012

Kwak 2008). Recent methods attempt to mitigate this sensitivity by adopting different error functions (He et al. 2011; Ding et al. 2006; Kwak 2008; Ke and Kanade 2003, 2005; Candés et al. 2009; de la Torre and Black 2003) which, however, often result in loss of efficiency. A reformulation of PCA in the context of M-Estimation is introduced in de la Torre and Black (2003). In Candés et al. (2009), PCA is represented as an optimization problem which finds a low dimensional linear subspace and a sparse matrix which represents the outliers. Other approaches in the literature use variants of the ℓ_1-norm which are, in general, more robust than the ℓ_2-norm (Ke and Kanade 2003, 2005; Ding et al. 2006; Kwak 2008; Mei and Ling 2009). More specifically, in Ke and Kanade (2003) and Ke and Kanade (2005) the optimization problem is considered with the following error function

$$\varphi(\mathbf{B}) = \left\|\mathbf{X} - \mathbf{B}\mathbf{B}^T\mathbf{X}\right\|_1. \tag{2}$$

The proposed ℓ_1-norm minimization is based on (i) the weighted median algorithm and (ii) convex quadratic programming, respectively. While this approach reduces the effect of outliers, the optimization of (2) is computationally expensive. Moreover, both methods are not invariant to rotations, which is an important property of learning algorithms (Ding et al. 2006).

The rotationally invariant R1-PCA (Ding et al. 2006) is also based on the ℓ_1-norm PCA

$$\varphi(\mathbf{B}) = \sum_{j=1}^{n}\sqrt{\sum_{c=1}^{p}\left(\mathbf{x}_j(c) - \sum_{l=1}^{m}\mathbf{b}_l(c)\sum_{r=1}^{p}\mathbf{b}_l(r)\mathbf{x}_j(r)\right)^2}, \tag{3}$$

where $\mathbf{x}_j(c)$ is the cth element of $\mathbf{x}_j$. PCA-L1 (Kwak 2008) estimates the optimum of (2) with a componentwise greedy search for

$$\mathbf{B} = \arg\max_{\check{\mathbf{B}}}\left\|\check{\mathbf{B}}^T\mathbf{X}\right\|_1. \tag{4}$$

Both methods allow for faster convergence towards the solution. Furthermore, both are rotational invariant. Most recently, Half-Quadratic PCA (HQ-PCA) is introduced in He et al. (2011). Here, the authors propose a rotational invariant and robust PCA using the maximum correntropy criterion (MCC) (Liu et al. 2007). Correntropy is closely related to M-Estimators while the objective function is efficiently optimized by the half-quadratic optimization technique. In contrast to the proposed method, incremental implementations of the above methods are unknown and therefore they are computationally expensive for large training sets and unsuitable for online learning.

Some of the problems often associated with robustness in PCA might be solved by more flexible modeling using Kernel PCA (KPCA) and more specifically by de-noising in feature spaces *via* the use of pre-images (Kwok and Tsang 2004; Mika et al. 1999; Honeine and Richard 2011). The problem of sample de-noising in feature space is formulated as follows. Let $k(.,.)$ be a positive definite function, the so-called kernel, and an implicit mapping ϕ, associated with the kernel, from the input space to a (possible) infinite dimensional Hilbert space. KPCA learns a linear subspace $\mathbf{B}$ in this high dimensional space. Then, a sample $\mathbf{x}$ is de-noised by solving the following optimization problem (Mika et al. 1999)

$$\tilde{\mathbf{x}} = \arg\min_{\check{\mathbf{x}}}\left\|\phi(\check{\mathbf{x}}) - \mathbf{B}\mathbf{B}^T\phi(\mathbf{x})\right\|_F^2. \tag{5}$$

Put in simple terms, the above optimization problem aims to find a sample $\tilde{\mathbf{x}}$ in input space such that its mapping in the feature space $\phi(\tilde{\mathbf{x}})$ approximates the reconstruction in the feature space $\mathbf{B}\mathbf{B}^T\phi(\mathbf{x})$ optimally. In Mika et al. (1999), a gradient descent methodology was proposed for solving optimization problem (5). Furthermore, fixed point algorithms were proposed for the case of isotropic kernels (*i.e.* kernels of the form $k(\mathbf{x}_j, \mathbf{x}_q) = f(\|\mathbf{x}_j - \mathbf{x}_q\|)$). Nevertheless, popular kernels, such as Gaussian Radial Basis Function (GRBF), do not posses robust properties by definition. To the best of our knowledge, the only method to address robust subspace estimation in feature spaces is the method in Nguyen and de la Torre (2009).

In this paper, we propose a KPCA with a kernel which has a direct connection to robust estimation as pointed out in Fitch et al. (2005). Our method is based on a dissimilarity measure which is originally introduced in Fitch et al. (2005) in the context of robust correlation-based estimation of large translational displacements. For this measure, pixel intensities are first normalized and, then, mapped onto the unit sphere using the Euler representation of complex numbers. Then, the standard ℓ_2-norm is applied. Overall, this is equivalent to applying a dissimilarity measure which is given by the cosine of the pixel differences. Note, the mapping is explicit and thus the proposed kernel PCA is closely related to standard PCA and retains all the favorable properties (e.g. efficiency and rotational invariance). Furthermore, it offers a very efficient approximation of pre-images without solving a separate optimization problem. Due to the existence of an explicit mapping to feature space, without increasing the dimensionality, it allows for an efficient incremental implementation. Incremental PCA is known to be more efficient than batch PCA when applied to large training sets (Li 2004; Ross et al. 2008). Furthermore, incremental PCA is more suitable for online learning. Overall, the proposed Euler-PCA (*e*-PCA) forms a fast, direct and robust alternative to standard PCA. We evaluate the performance of our method on several computer vision problems often found in practical Human Computer Interaction (HCI) systems (Oliver et al. 2000; Wren et al. 1997): face reconstruction, tracking and background modeling for change detection. Summarizing the favorable properties of the proposed Euler-PCA are

– contrary to the state-of-the-art linear robust PCA and KPCA approaches (Candés et al. 2009; He et al. 2011;

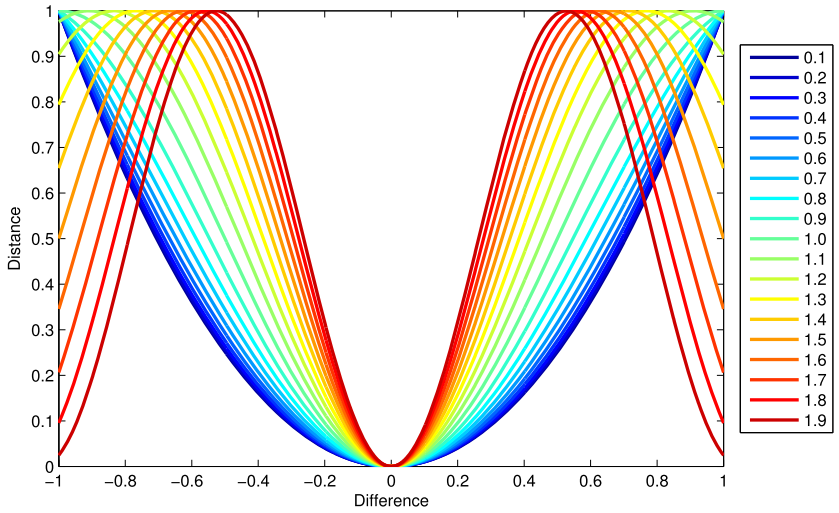

Fig. 1 The cosine dissimilarity measure with changing α. (The distance is normalized for illustration purposes)

Ding et al. 2006; Kwak 2008; Ke and Kanade 2003, 2005; de la Torre and Black 2003; Chin et al. 2006) Euler-PCA allows for an efficient incremental implementation
– contrary to KPCA approaches with standard kernels such as GRBF we show that there exists an efficient way for pre-image computation without solving optimization problems.

In the experiments we show that the proposed Euler-PCA not only possesses these favorable properties but also outperforms the state-of-the-art.

Finally we note that compared to Tzimiropoulos (2012), our proposed method neither relies on the statistics of the gradient orientation differences nor restricts itself to the domain of gradient orientations in general. Our work is a learning method directly derived from the correlation method of Fitch et al. (2005), while Tzimiropoulos (2012) can be seen as the learning method derived from the analysis of the orientation correlation function presented in Tzimiropoulos (2010).

The rest of the paper is organized as follows. In Sect. 2 we introduce the cosine-based dissimilarity measure that forms the basis of the proposed Euler-PCA. In Sect. 3 we formulate the proposed Euler-PCA and discuss some of its properties. Experimental results are presented in Sect. 4. Finally, conclusions are drawn in Sect. 5. Video sequences of our results are provided in the supplementary material.

2 Cosine-Based Dissimilarity

Let $\mathbf{x}_j$ be the p-dimensional vector obtained by writing image I_j in lexicographic ordering. Motivated by the recent work in Fitch et al. (2005) on robust correlation-based translation estimation, we replace the ℓ_2-norm with the following dissimilarity measure

$$d(\mathbf{x}_j, \mathbf{x}_q) = \sum_{c=1}^{p} \left\{1 - \cos\left(\alpha\pi\left(\mathbf{x}_j(c) - \mathbf{x}_q(c)\right)\right)\right\}, \tag{6}$$

where the pixel values of the corresponding images I_j, I_q are represented in the range $[0, 1]$ and $\alpha \in \mathbb{R}^+$. Figure 1 visualizes the dissimilarity function with changing values for α. A small value for α results in a function which resembles the ℓ_2-norm. With increasing α, the effect of large distances possibly caused by outliers is reduced. In general, α represents the frequency of the cosine and is optimized to suppress the values caused by outliers.

As noted in Fitch et al. (2005), for pixel intensities in the range $[0, 1]$, (6) is equivalent to Andrews' M-Estimate. In particular, the influence function of the kernel, *i.e.* the derivative of the kernel, is equivalent to Andrews' influence function, which is given by

$$\psi(r) = \begin{cases} \sin(\pi r) & \text{if } -1 \le r \le 1 \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

The Fast Robust Correlation (FRC) scheme in Fitch et al. (2005) utilizes (6) and, unlike ℓ_2-based correlation, is able to estimate large translational displacements in real images while achieving the same computational complexity.

Prior to formulating the proposed PCA, let us consider a motivating example in which different dissimilarity measures are applied to the images shown in Fig. 2. As can be seen in Table 1, the ℓ_2-norm associates a smaller distance between the original image and an image from a different subject. The distance between the original and the same image with occlusion is larger. In contrast, the use of the cosine-based measure[2] results in a large distance between the original image and the image of a different person.

[2] We set $\alpha = 1.9$, as will be discussed later, in Sect. 4.

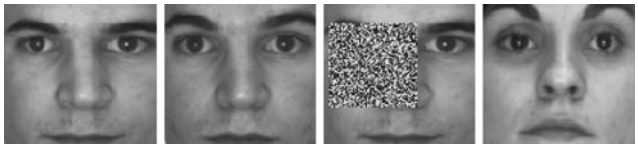

Fig. 2 Example motivating the use of the cosine-based dissimilarity measure. Shown from left to right are the original image, a second image of the same subject, an occluded version of the original image and an image of another subject

Table 1 Comparison of normalized dissimilarity measures

	ℓ_2-Norm	Cosine measure
Same subject	0.979×10^{-3}	13.404×10^{-3}
Occluded	1.96×10^{-3}	33.576×10^{-3}
Different subject	1.63×10^{-3}	34.599×10^{-3}

3 Euler-PCA (*e*-PCA)

3.1 Batch Version

In this section we first present Euler-PCA and introduce its representation as both KPCA and linear PCA with special features.

Euler-PCA is a KPCA which utilizes the robust dissimilarity in (6). It is also based on the Euler representation of complex numbers. More specifically, we map the intensity values $\mathbf{x}_j$ normalized in $[0, 1]$ onto the complex representation $\mathbf{z}_j \in \mathbb{C}^p$, where

$$\mathbf{z}_j = \frac{1}{\sqrt{2}} \begin{bmatrix} e^{i\alpha\pi \mathbf{x}_j(1)} \\ \vdots \\ e^{i\alpha\pi \mathbf{x}_j(p)} \end{bmatrix} = \frac{1}{\sqrt{2}} e^{i\alpha\pi \mathbf{x}_j}. \tag{8}$$

The values $\mathbf{z}_j$ can be thought of as the special features in our version of PCA. To show the relationship between (8) and (6), let us define $\boldsymbol{\theta}_j \triangleq \alpha\pi\mathbf{x}_j$ and then apply the ℓ_2-norm

$$\begin{aligned} \|\mathbf{z}_j - \mathbf{z}_q\|_F^2 &= \frac{1}{2} \big\| \big(\cos(\alpha\pi\mathbf{x}_j) + i\sin(\alpha\pi\mathbf{x}_j)\big) \\ &\quad - \big(\cos(\alpha\pi\mathbf{x}_q) + i\sin(\alpha\pi\mathbf{x}_q)\big) \big\|_F^2 \\ &= \sum_{c=1}^{p} \big\{1 - \cos\big(\boldsymbol{\theta}_j(c) - \boldsymbol{\theta}_q(c)\big)\big\} \\ &= d(\mathbf{x}_j, \mathbf{x}_q). \end{aligned} \tag{9}$$

This suggests that our KPCA can be defined by first applying the *explicit* mapping of (8) and then using standard linear complex PCA. Because of (8), we coin this approach as Euler-PCA. Notice that for $\alpha < 2$, this mapping is one-to-one. In this case, this gives rise to fast pre-image computations *via* the $\angle$-operator, which returns the angle of a complex number. More specifically, after reconstruction (or de-noising) in the feature space has been performed, we can go back to the pixel domain using the $\angle$-operator. Finally, high-dimensional data can be readily handled by using Theorem 1 (a proof can be found in Appendix A).

Algorithm 1 ESTIMATING THE PRINCIPAL SUBSPACE

Input: A set of n images I_j, $j = 1, \ldots, n$, of p pixels, the number m of principal components and parameter α.

Output: The principal subspace $\mathbf{B}$ and eigenvalues $\boldsymbol{\Sigma}$.

1: Represent I_j in the range $[0, 1]$ and obtain $\mathbf{x}_j$ by writing I_j in lexicographic ordering.
2: Compute $\mathbf{z}_j$ using (8) and form the matrix of the transformed data $\mathbf{Z} = [\mathbf{z}_1 \cdots \mathbf{z}_n] \in \mathbb{C}^{p\times n}$.
3: Compute the kernel matrix $\mathbf{K} = \mathbf{Z}^H\mathbf{Z} \in \mathbb{C}^{n\times n}$ and find the eigendecomposition of $\mathbf{K} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^H$.
4: Find the m-reduced set, $\mathbf{U}_m \in \mathbb{C}^{n\times m}$ and $\boldsymbol{\Lambda}_m \in \mathbb{R}^{m\times m}$.
5: Compute $\mathbf{B} = \mathbf{Z}\mathbf{U}_m\boldsymbol{\Lambda}_m^{-\frac{1}{2}} \in \mathbb{C}^{p\times m}$ and $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}_m^{\frac{1}{2}}$.
6: Reconstruct using $\tilde{\mathbf{Z}} = \mathbf{B}\mathbf{B}^H\mathbf{Z}$.
7: Fast pre-image computation: go back to the pixel domain using $\tilde{\mathbf{X}} = \frac{\angle\tilde{\mathbf{Z}}}{\alpha\pi}$.

Algorithm 2 EMBEDDING OF NEW SAMPLES

Input: The principal subspace $\mathbf{B}$ and α of **Algorithm 1**, as well as a new image I of p pixels.

Output: The embedding in subspace and pixel domain, $\tilde{\mathbf{z}}$ and $\tilde{\mathbf{x}}$ respectively.

1: Represent I in the range $[0, 1]$ and obtain $\mathbf{x}$ by writing I in lexicographic ordering.
2: Find $\mathbf{z}$ using (8) and reconstruct as $\tilde{\mathbf{z}} = \mathbf{B}\mathbf{B}^H\mathbf{z}$.
3: Fast pre-image computation: go back to the pixel domain using $\tilde{\mathbf{x}} = \frac{\angle\tilde{\mathbf{z}}}{\alpha\pi}$.

Theorem 1 *Define matrices* $\mathbf{A}$ *and* $\mathbf{B}$ *such that* $\mathbf{A} = \boldsymbol{\Phi}\boldsymbol{\Phi}^H$ *and* $\mathbf{B} = \boldsymbol{\Phi}^H\boldsymbol{\Phi}$, *where* $.^H$ *computes the complex conjugate transposition of a matrix. Let* $\mathbf{U}_A$ *and* $\mathbf{U}_B$ *be the eigenvectors corresponding to the non-zero eigenvalues* $\boldsymbol{\Lambda}_A$ *and* $\boldsymbol{\Lambda}_B$ *of* $\mathbf{A}$ *and* $\mathbf{B}$, *respectively. Then,* $\boldsymbol{\Lambda}_A = \boldsymbol{\Lambda}_B$ *and* $\mathbf{U}_A = \boldsymbol{\Phi}\mathbf{U}_B\boldsymbol{\Lambda}_A^{-\frac{1}{2}}$.

The complete Euler-PCA is given in Algorithm 1 and Algorithm 2.

Let us now discuss Euler-PCA's interpretation as a KPCA with a complex kernel in an explicitly defined complex Hilbert space. Let $k : \mathbb{R}^p \times \mathbb{R}^p \to \mathbb{C}$ be a positive definite function that defines a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H}$ (the so-called feature space) through an implicit (or as in our case explicit) mapping $\phi : \mathbb{R}^p \to \mathcal{H}$ such that $k(\mathbf{x}_j, \mathbf{x}_q) = \langle\phi(\mathbf{x}_j), \phi(\mathbf{x}_q)\rangle$ (Paulsen 2009). The inner product is given by $\phi(\mathbf{x}_j)^H\phi(\mathbf{x}_q)$, with the property $k(\mathbf{x}_j, \mathbf{x}_q) = \overline{k(\mathbf{x}_q, \mathbf{x}_j)}$ (where $\bar{.}$ is the complex conjugate operator). Using the complex feature space interpretation the kernel that corresponds to the mapping (8) can be written as

$$k(\mathbf{x}_j,\mathbf{x}_q)=\frac{1}{2}\sum_{c=1}^{p}\cos\big(\alpha\pi\left(\mathbf{x}_j(c)-\mathbf{x}_q(c)\right)\big)$$
$$-\,i\frac{1}{2}\sum_{c=1}^{p}\sin\big(\alpha\pi\left(\mathbf{x}_j(c)-\mathbf{x}_q(c)\right)\big). \tag{10}$$

With the KPCA interpretation of the proposed method the strategy for reconstruction in the input space becomes less apparent. In the following section, we present the reconstruction by means of pre-image computation (Kwok and Tsang 2004; Mika et al. 1999) (a recent survey on pre-image computation problems can be found in Honeine and Richard (2011)). Then, we put our suggested approximation $\tilde{\mathbf{x}}=\frac{\angle\tilde{\mathbf{z}}}{\alpha\pi}$ of Algorithms 1 and 2 for going back to the pixel domain (input space) in the context of pre-image computation and we derive a closed form to the approximation error.

3.2 Pre-image Computation

The optimization problem (5) for the proposed kernel can be reformulated as

$$\begin{aligned}\tilde{\mathbf{x}}&=\arg\min_{\check{\mathbf{x}}}\left\|\phi(\check{\mathbf{x}})-\mathbf{B}\mathbf{B}^H\phi(\mathbf{x})\right\|_F^2\\&=\arg\min_{\check{\mathbf{x}}}\big\{k(\check{\mathbf{x}},\check{\mathbf{x}})+\phi(\mathbf{x})^H\mathbf{B}\mathbf{B}^H\mathbf{B}\mathbf{B}^H\phi(\mathbf{x})\\&\quad-\phi(\check{\mathbf{x}})^H\mathbf{B}\mathbf{B}^H\phi(\mathbf{x})-\phi(\mathbf{x})^H\mathbf{B}\mathbf{B}^H\phi(\check{\mathbf{x}})\big\}\\&=\arg\min_{\check{\mathbf{x}}}\big\{-\phi(\check{\mathbf{x}})^H\mathbf{B}\mathbf{B}^H\phi(\mathbf{x})-\phi(\mathbf{x})^H\mathbf{B}\mathbf{B}^H\phi(\check{\mathbf{x}})\big\}\\&=\arg\max_{\check{\mathbf{x}}}\mathtt{Re}\big(\phi(\check{\mathbf{x}})^H\mathbf{B}\mathbf{B}^H\phi(\mathbf{x})\big),\end{aligned} \tag{11}$$

where $\mathtt{Re}(.)$ extracts the real part of a complex number and $\mathtt{Im}(.)$ the corresponding imaginary. Note, in KPCA the projection matrix is represented as a linear combination $\mathbf{B}=\mathbf{X}^{\phi}\tilde{\mathbf{B}}$, where $\mathbf{X}^{\phi}=[\phi(\mathbf{x}_1)\cdots\phi(\mathbf{x}_n)]$. Then, by setting

$$\mathbf{t}=\tilde{\mathbf{B}}\tilde{\mathbf{B}}^H\begin{bmatrix}k(\mathbf{x}_1,\mathbf{x})\\\vdots\\k(\mathbf{x}_N,\mathbf{x})\end{bmatrix}$$

the optimization problem (11) can be reformulated as

$$\begin{aligned}\tilde{\mathbf{x}}&=\arg\max_{\check{\mathbf{x}}}\mathtt{Re}\big(\big[k(\check{\mathbf{x}},\mathbf{x}_1)\ldots k(\check{\mathbf{x}},\mathbf{x}_n)\big]\mathbf{t}\big)\\&=\arg\max_{\check{\mathbf{x}}}\big\{\mathtt{Re}\big(\big[k(\check{\mathbf{x}},\mathbf{x}_1)\ldots k(\check{\mathbf{x}},\mathbf{x}_n)\big]\big)\mathtt{Re}(\mathbf{t})\\&\quad-\mathtt{Im}\big(\big[k(\check{\mathbf{x}},\mathbf{x}_1)\ldots k(\check{\mathbf{x}},\mathbf{x}_n)\big]\big)\mathtt{Im}(\mathbf{t})\big\}\\&=\arg\max_{\check{\mathbf{x}}}\sum_{j=1}^{n}\sum_{c=1}^{p}\big\{\cos\big(\alpha\pi\left(\check{\mathbf{x}}(c)-\mathbf{x}_j(c)\right)\big)\mathtt{Re}\big(\mathbf{t}(j)\big)\\&\quad+\sin\big(\alpha\pi\left(\check{\mathbf{x}}(c)-\mathbf{x}_j(c)\right)\big)\mathtt{Im}\big(\mathbf{t}(j)\big)\big\}\\&=\arg\max_{\check{\mathbf{x}}}f(\check{\mathbf{x}}).\end{aligned} \tag{12}$$

The standard way to optimize (12) is by gradient ascent (*i.e.* an update of the form $\check{\mathbf{x}}_t=\check{\mathbf{x}}_{t-1}+\nabla f(\check{\mathbf{x}}_{t-1})$ (Mika et al. 1999)). Hence, we need to compute the partial derivatives $\frac{\partial f}{\partial\check{\mathbf{x}}(c)}$ for all pixels as

$$\frac{\partial f}{\partial\check{\mathbf{x}}(c)}=-\alpha\pi\sum_{j=1}^{N}\mathtt{Re}\big(\mathbf{t}(j)\big)\sin\big(\alpha\pi\left(\check{\mathbf{x}}(c)-\mathbf{x}_j(c)\right)\big)$$
$$+\alpha\pi\sum_{j=1}^{N}\mathtt{Im}\big(\mathbf{t}(j)\big)\cos\big(\alpha\pi\left(\check{\mathbf{x}}(c)-\mathbf{x}_j(c)\right)\big). \tag{13}$$

Using (13), ∇f can be concisely written as

$$\nabla f(\check{\mathbf{x}})=-\big[\mathtt{Im}(\bar{\check{\mathbf{z}}}\odot\mathbf{z}_1)\ldots\mathtt{Im}(\bar{\check{\mathbf{z}}}\odot\mathbf{z}_n)\big]\begin{bmatrix}\mathtt{Re}(\mathbf{t}(1))\\\vdots\\\mathtt{Re}(\mathbf{t}(n))\end{bmatrix}$$
$$+\big[\mathtt{Re}(\bar{\check{\mathbf{z}}}\odot\mathbf{z}_1)\ldots\mathtt{Re}(\bar{\check{\mathbf{z}}}\odot\mathbf{z}_n)\big]\begin{bmatrix}\mathtt{Im}(\mathbf{t}(1))\\\vdots\\\mathtt{Im}(\mathbf{t}(n))\end{bmatrix}, \tag{14}$$

where $\odot$ is the element-wise product between vectors, $\check{\mathbf{z}}$ is the transform $\phi(\check{\mathbf{x}})$ in (8) and $\bar{.}$ computes the complex conjugate of a vector $\mathbf{x}$. Unfortunately, the above procedure can be quite computational expensive for large databases (the order for recovering K pre-images is $O(\mu pKn)$ where μ is the number of steps until convergence).

In contrast, Algorithm 1 and 2 we have proposed a very fast way for approximating pre-images by

$$\tilde{\mathbf{x}}=\frac{1}{\pi\alpha}\angle\mathbf{B}\mathbf{B}^H\mathbf{z}. \tag{15}$$

In the context of pre-image computation the error of this approximation can $\tilde{\mathbf{x}}$ can be analytically computed by substituting (15) into (8) and the result into (5) (details can be found in Appendix B)

$$\left\|\phi(\check{\mathbf{x}})-\mathbf{B}\mathbf{B}^H\phi(\mathbf{x})\right\|_F^2=\left\|\frac{1}{\sqrt{2}}e^{i\angle\mathbf{B}\mathbf{B}^H\mathbf{z}}-\mathbf{B}\mathbf{B}^H\mathbf{z}\right\|_F^2$$
$$=\left\|\frac{1}{\sqrt{2}}\mathbf{1}-\mathtt{R}\big(\mathbf{B}\mathbf{B}^H\mathbf{z}\big)\right\|_F^2, \tag{16}$$

where $\mathtt{R}(\mathbf{b})=[\sqrt{\mathtt{Re}(\mathbf{b}(c))^2+\mathtt{Im}(\mathbf{b}(c))^2}]$ is a vector containing the magnitude of the elements in $\mathbf{b}$ and $\mathbf{1}$ is a vector of ones. Finally, due to the invertibility of the proposed mapping (8) for $0\leq\alpha<2$, in the case of $\mathbf{B}$ containing all eigenvectors that correspond to non-zero eigenvalues, it can easily be shown that the pre-image approximation using (15) is optimal for the training set (*i.e.* (16) is equal to zero).

3.3 Incremental Learning

We base the update method for the incremental Euler-PCA on standard incremental PCA (Ross et al. 2008; Levy and Lindenbaum 2000), as Euler-PCA is formulated as linear

Algorithm 3 INCREMENTAL SUBSPACE ESTIMATION

Input: The principal subspace $\mathbf{B}_{t-1} \in \mathbb{C}^{p\times m}$, the corresponding eigenvalues $\boldsymbol{\Sigma}_{t-1} \in \mathbb{R}^{m\times m}$, a set of new images $\{I_{n+1}, \ldots, I_{n+a}\}$, the number m of principal components and parameter α.

Output: The new subspace $\mathbf{B}_t$ and eigenvalues $\boldsymbol{\Sigma}_t$.

1: From set $\{I_{n+1}, \ldots, I_{n+a}\}$ compute the matrix of the transformed data $\mathbf{Z}_\delta = [\mathbf{z}_{n+1} \cdots \mathbf{z}_{n+a}]$.

2: Compute $\mathbf{B}_\delta = \mathtt{orth}(\mathbf{Z}_\delta - \mathbf{B}_{t-1}\mathbf{B}_{t-1}^H\mathbf{Z}_\delta)$ and form $\mathbf{L} = \begin{bmatrix} \boldsymbol{\Sigma}_{t-1} & \mathbf{B}_{t-1}{}^H\mathbf{Z}_\delta \\ \mathbf{0} & \mathbf{B}_\delta^H\mathbf{Z}_\delta \end{bmatrix}$.

3: Compute $\mathbf{L} \overset{svd}{=} \hat{\mathbf{B}}\hat{\boldsymbol{\Sigma}}\hat{\mathbf{V}}^H$ and obtain the m-reduced set $\hat{\mathbf{B}}_m$ and $\hat{\boldsymbol{\Sigma}}_m$.

4: Compute $\mathbf{B}_t = [\mathbf{B}_{t-1}\ \mathbf{B}_\delta]\hat{\mathbf{B}}_m$ and set $\boldsymbol{\Sigma}_t = \hat{\boldsymbol{\Sigma}}_m$.

PCA in a non-linear subspace defined by an explicit mapping. We assume that the subspace $\mathbf{B}_{t-1}$ of step $t-1$ is given by the SVD of $\mathbf{Z}_{t-1}$

$$\mathbf{Z}_{t-1} \approx \mathbf{B}_{t-1}\boldsymbol{\Sigma}_{t-1}\mathbf{V}_{t-1}^H = \mathbf{Z}_{t-1}\mathbf{U}_m\boldsymbol{\Lambda}_m^{-\frac{1}{2}}\boldsymbol{\Lambda}_m^{\frac{1}{2}}\mathbf{U}_m^H, \tag{17}$$

where $\mathbf{U}_m\boldsymbol{\Lambda}_m\mathbf{U}_m^H$ is the m-reduced eigenvalue decomposition of $\mathbf{Z}_{t-1}^H\mathbf{Z}_{t-1}$. For the update, we want to find the SVD of the concatenated sample matrix, build from $\mathbf{Z}_{t-1}$, and the new mapped samples $\mathbf{Z}_\delta$,

$$\mathbf{Z}_t = [\mathbf{Z}_t \quad \mathbf{Z}_\delta] \approx \left[\mathbf{B}_{t-1}\boldsymbol{\Sigma}_{t-1}\mathbf{V}_{t-1}^H \quad \mathbf{Z}_\delta\right]. \tag{18}$$

We reformulate (18) as

$$\mathbf{Z}_t \approx [\mathbf{B}_{t-1} \quad \mathbf{B}_\delta]\begin{bmatrix} \boldsymbol{\Sigma}_{t-1} & \mathbf{B}_{t-1}^H\mathbf{Z}_\delta \\ \mathbf{0} & \mathbf{B}_\delta^H\mathbf{Z} \end{bmatrix}\begin{bmatrix} \mathbf{V}_{t-1}^H & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}, \tag{19}$$

where $\mathbf{B}_\delta = \mathtt{orth}(\mathbf{Z}_\delta - \mathbf{B}\mathbf{B}^H\mathbf{Z}_\delta)$ contains the new components, which are not included in the current subspace $\mathbf{B}_{t-1}$. Finally, the SVD of

$$\mathbf{L} = \begin{bmatrix} \boldsymbol{\Sigma}_{t-1} & \mathbf{B}_{t-1}^H\mathbf{Z}_\delta \\ \mathbf{0} & \mathbf{B}_\delta^H\mathbf{Z} \end{bmatrix} = \hat{\mathbf{B}}\hat{\boldsymbol{\Sigma}}\hat{\mathbf{V}}^H \tag{20}$$

is all that is required for the update. With the m-reduced eigenspace $\hat{\mathbf{B}}_m$ and $\hat{\boldsymbol{\Sigma}}_m$ of $\mathbf{L}$, we find $\mathbf{B}_t = [\mathbf{B}_{t-1} \quad \mathbf{B}_\delta]\hat{\mathbf{B}}_m$ and $\boldsymbol{\Sigma} = \hat{\boldsymbol{\Sigma}}_m$. Algorithm 3 summarizes the main steps.

Note that existing methods for incremental KPCA in which the mapping is in general unknown are computationally expensive and inexact. For example in Chin and Suter (2007), to ensure constant execution speed, a set of pre-images are found to approximate the data matrix by solving an extra optimization problem similar to (5). The drawbacks of this are twofold: (i) the reduced set representation provides only an estimate to the exact solution and (ii) the proposed optimization problem for finding the reduced set inevitably increases the complexity of the algorithm. On the other hand, since in our case, the mapping is explicit and does not increase the dimensionality we can represent the data matrix directly in feature space. This eliminates the need to introduce an additional optimization problem, making the incremental version of Euler-PCA both fast and exact.

Fig. 3 Cropped example images from the AR Database (Martinez and Benavente 1998)

4 Experiments

4.1 Image Reconstruction Under Noise

In this section we evaluate the robustness of Euler-PCA (e-PCA) for the application on image de-noising based on subspace-based image reconstruction. For comparison, we select standard PCA, R1-PCA (Ding et al. 2006), PCA-L1 (Kwak 2008) and HQ-PCA (He et al. 2011), which represents the state-of-the-art, as well as, standard Kernel PCA de-noising with a GRBF KPCA (denoted by G-KPCA) and pre-image computation using (15) (denoted by e-PCA-GA). The parameters of R1-PCA, PCA-L1 and HQ-PCA follow (Ding et al. 2006; Kwak 2008; He et al. 2011) respectively. We choose the convergence criterion for R1-PCA, PCA-L1 and HQ-PCA to be based on the norm difference between two successive subspace estimations. The maximum difference is constrained not to exceed 10^{-8}, unless a maximum of 50 iterations is reached. For the optimization of G-KPCA variance of the Gaussian kernel we tried two standard approaches, but for compactness we always report the best results. In the first approach we set the variance equal to $\frac{1}{n(n-1)}\sum_{j,q}^{n}\|\mathbf{x}_j - \mathbf{x}_q\|^2$ where $\mathbf{x}_j$, $\mathbf{x}_q$ are the training samples (Kwok and Tsang 2004). In the second method we applied a cross-validation strategy in the training set for selecting the variance. For all methods that employ a gradient descent (or ascent) for pre-image computation the pre-images were initialized with all pixel intensities equal to 0.5. Finally, the methods were considered to have converged if $\|\check{\mathbf{x}}_t - \check{\mathbf{x}}_{t-1}\|_F \leq 0.01$ for two successive iterations t and $t-1$.

Our data set consists of a subset of the popular AR Database (Martinez and Benavente 1998). In particular, we use a total of 100 images of size 101×91 of different subjects as shown in Fig. 3.

Our evaluation is based on the reconstruction error and the angular error (He et al. 2011; Kwak 2008; Gunawan et al. 2005; Krzanowski 1979). The reconstruction error has been used in many evaluations of previous approaches (He et al.

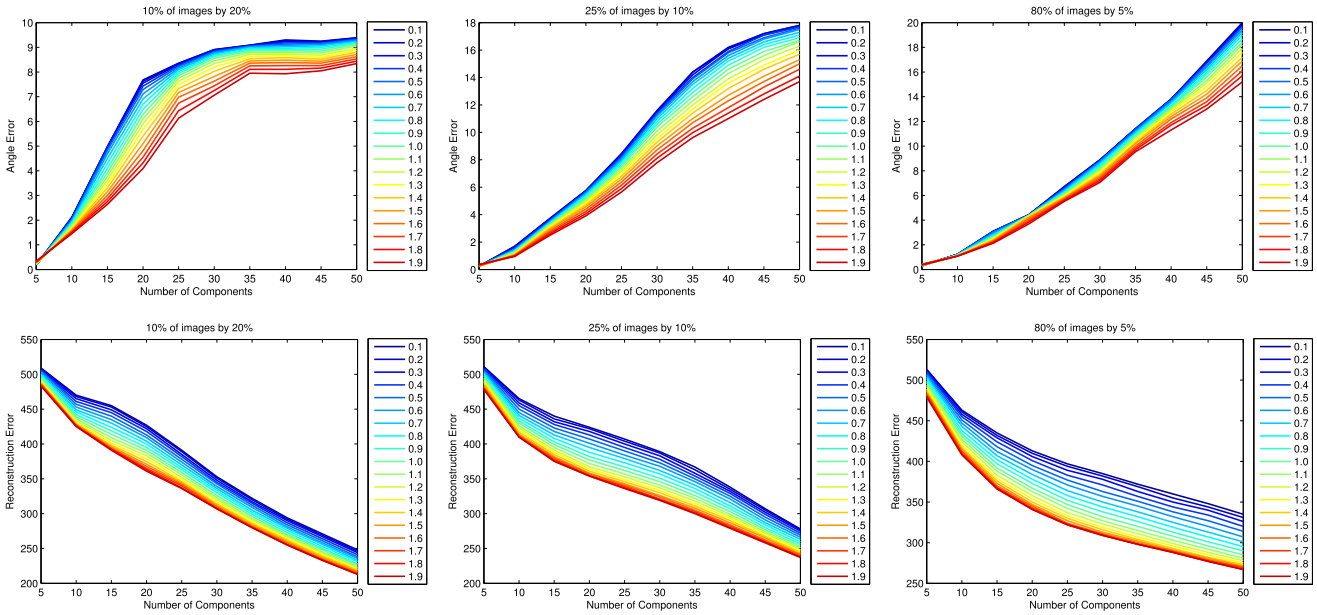

Fig. 4 Angular error (*top*) and reconstruction error (*bottom*) for different values for α

2011; Kwak 2008). In particular, for n test samples, we compute

$$e_r(m) = \frac{1}{n}\sum_{j=1}^{n}\left\|\mathbf{x}_j^{orig} - \sum_{l=1}^{m}\mathbf{b}_l^{cor}\mathbf{b}_l^{cor^T}\mathbf{x}_j^{cor}\right\| \tag{21}$$

where $\mathbf{x}_j^{orig}$ and $\mathbf{x}_j^{cor}$ represent the original and the training image respectively, $\mathbf{B}^{cor} = [\mathbf{b}_1^{cur}\cdots\mathbf{b}_m^{cur}] \in \mathbb{R}^{p\times m}$ is the estimated subspace of the corrupted data and m denotes the number of components used. For the methods that use pre-images for approximating the reconstruction error (21) is reformulated as

$$e_r(m) = \frac{1}{n}\sum_{j=1}^{n}\left\|\mathbf{x}_j^{orig} - \tilde{\mathbf{x}}_j^{cor}(m)\right\| \tag{22}$$

where $\tilde{\mathbf{x}}_j^{cor}(m)$ is the pre-image associated with the reconstruction using m components in the feature space (*i.e.* solving optimization problems (5) and (11) using m components for matrix $\mathbf{B}$). Note, that for Euler-PCA de-noising is performed by calculating pre-images in two ways: by applying the $\angle$-operator and by the gradient ascent optimization. We denote the latter method as e-PCA-GA. The calculation of pre-images for G-KPCA is also performed in a similar fashion.

Additionally we use the angular error between the corrupted subspace $\mathbf{B}^{cor}$ (learned from the corrupted training set) and the uncorrupted subspace $\mathbf{B}^{orig} = [\mathbf{b}_1^{orig}\cdots\mathbf{b}_m^{orig}] \in \mathbb{R}^{p\times m}$ (learned from the original images) as follows (Gunawan et al. 2005; Krzanowski 1979)

$$e_a(m) = m - \sum_{l=1}^{m}\sum_{s=1}^{m}\cos^2\left(\mathbf{b}_l^{orig}, \mathbf{b}_s^{cor}\right). \tag{23}$$

For the nonlinear methods $\mathbf{B}^{orig}$ and $\mathbf{B}^{cor}$ are in the feature space but still $\cos(\mathbf{b}_l^{orig}, \mathbf{b}_s^{cor})$ can be efficiently computed using the kernel. In contrast to the reconstruction error which introduces an inherent error due to the chosen number of components, the angle error shows the difference caused by the outliers directly.

4.1.1 Synthetic Corruptions

In this experiment, a percentage of training images is corrupted by randomly placed patches of random pixel noise as shown in Fig. 9. We vary both the number of corrupted images and the size of the corrupted area. For convenience, we say "*a* % of images by *b* %" to denote that *a* % of the images in our training set was corrupted by randomly placed patches the size of which is *b* % of the total image size. After training we analyze the results based on the reconstruction and angle errors.

In our experiments, we tested the 5 different methods for 7 setups. These setups can be summarized into three types:

- Type (i): *large occlusions on few images* (*e.g.* 5 % *of images by* 30 % *and* 10 % *of images by* 30 %)
- Type (ii): *medium sized occlusions on a few images* (*e.g.* 10 % *of images by* 20 %, 15 % *of images by* 10 % *and* 25 % *of images by* 10 %)
- Type (iii): *small occlusions on many images* (*e.g.* 80 % *of images by* 5 % *and* 85 % *of images by* 5 %)

Prior to our experiments, we optimized α of e-PCA *via* a validation set. We found that for $0 \leq \alpha < 2$ performance was attained for $\alpha = 1.9$. This also verifies the findings of Fig. 4. We have to note here that α was kept fixed to $\alpha = 1.9$

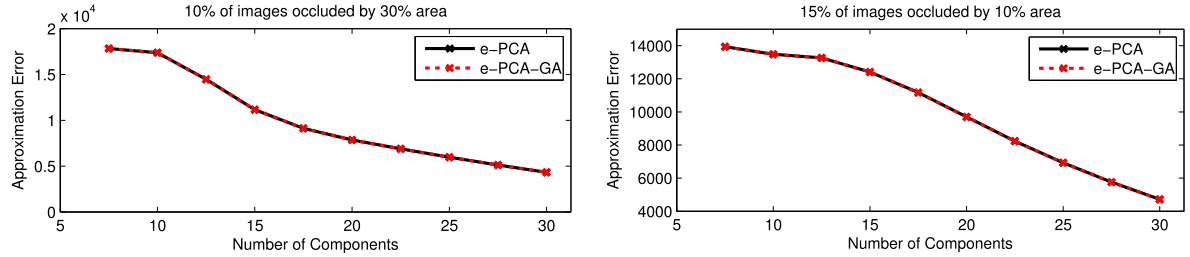

Fig. 5 Approximation error between *e*-PCA and *e*-PCA-GA

for all the experiments in this paper. This is in contrast with GKPCA which includes an extra step for finding the optimum variance.

In order to test whether the pre-image approximation using (15) is a valid choice we calculated the attained minimum of optimization problem (11), after performing the gradient ascent, and we compare it with the error given in (16). For all experiments the error of the fast pre-image approximation resulted in similar or lower errors. A representative example can be found in Fig. 5 where the pre-image approximation error is plotted versus the number of components. It is evident that both the pre-image computation methods produce similar errors (here we have to note that the pre-image approximation error is different than the reconstruction error in (22) since the former is in the feature space while the latter is in the input space).

Figure 6 shows the reconstruction errors of all tested methods. In type (i) and (ii), HQ-PCA performs well for few components, while R1-PCA performs worse than HQ-PCA but better than standard PCA. As the number of components increases, PCA, R1-PCA and HQ-PCA perform the same. PCA-L1 performs well only for a small number of components. G-KPCA performs as good as HQ-PCA. In all examples, both versions of the proposed Euler-PCA performs the best even for a large number of components.

More distinctions between the tested methods are observable for type (iii). Here, PCA and R1-PCA have similar performance, up to 30 components, after which R1-PCA outperforms standard PCA. Similar conclusions can be drawn for HQ-PCA. Again, Euler-PCA outperforms all other methods. Qualitative reconstruction results can be seen in Fig. 9. As it can be seen, our method is able to largely suppress such outliers.

The angular error results reveal different performance as it can be seen in Fig. 7. HQ-PCA outperforms PCA-L1 only for a large number of components. R1-PCA and standard PCA perform similarly. G-KPCA seems to perform the second best. This suggest that the performance improvement obtained for G-KPCA might be due to the fact that, for this experiment, the calculation of pre-images is not necessary. It seems that this calculation (as required by the reconstruction experiment) might be problematic. Once more, the proposed Euler-PCA performs best.

4.1.2 Hand Occlusions

In our second experiment we use skin-like occlusions to verify the results of the previous section. In particular, we occlude a subset of the training data with hand signs of the American fingerspelling alphabet[3] (Fig. 8). The chosen sign (letter), its orientation and its position are randomized, and the skin color is adjusted to fit the subject.

Figure 10 shows the reconstruction error and Fig. 11 the angular error. As before, HQ-PCA and G-KPCA outperform R1-PCA and standard PCA. Again, PCA-L1 performs the worst. Euler-PCA performs the best. Slightly different results can be observed for the angular error. In contrast to our previous setup, here, R1-PCA, PCA and Euler-PCA perform similarly well. Again, G-KPCA performs very well for this experiment. However, HQ-PCA and PCA-L1 seem to perform worse. The corruption by hand occlusions is much more subtle than the one introduced by the random pixel patches. Therefore, PCA and R1-PCA achieve a similar performance to Euler-PCA in terms of angular error. Nonetheless, the general trend follows our results of the previous section. Euler-PCA works well in terms of both reconstruction and angular error. Figure 12 shows an example of the reconstruction quality.

4.2 Object Tracking

We evaluate the incremental version of Euler-PCA for the application of visual tracking. The aim of a visual tracking system is to locate a predefined target object on every frame of a video sequence. Automatic systems span a wide range of applications, such as traffic monitoring (Kamijo et al. 2000; Hsieh et al. 2006), surveillance (Haritaoglu et al. 2000; Collins et al. 2001), video retrieval and summarization (Luo et al. 2003), vehicle navigation (Hashima et al. 1997; Fraundorfer et al. 2007), driver assistance (Handmann et al. 1998; Avidan 2004), human computer interaction (Wren et al. 1997; Liwicki and Everingham 2009) and face analysis (Gunes and Pantic 2010; Cohn et al. 1999). Many tracking algorithms indicate that an adaptive approach based on

[3]The fingerspelling alphabet is a subset of sign language which is utilized for spelling names. Examples can be found at http://asl.ms/.

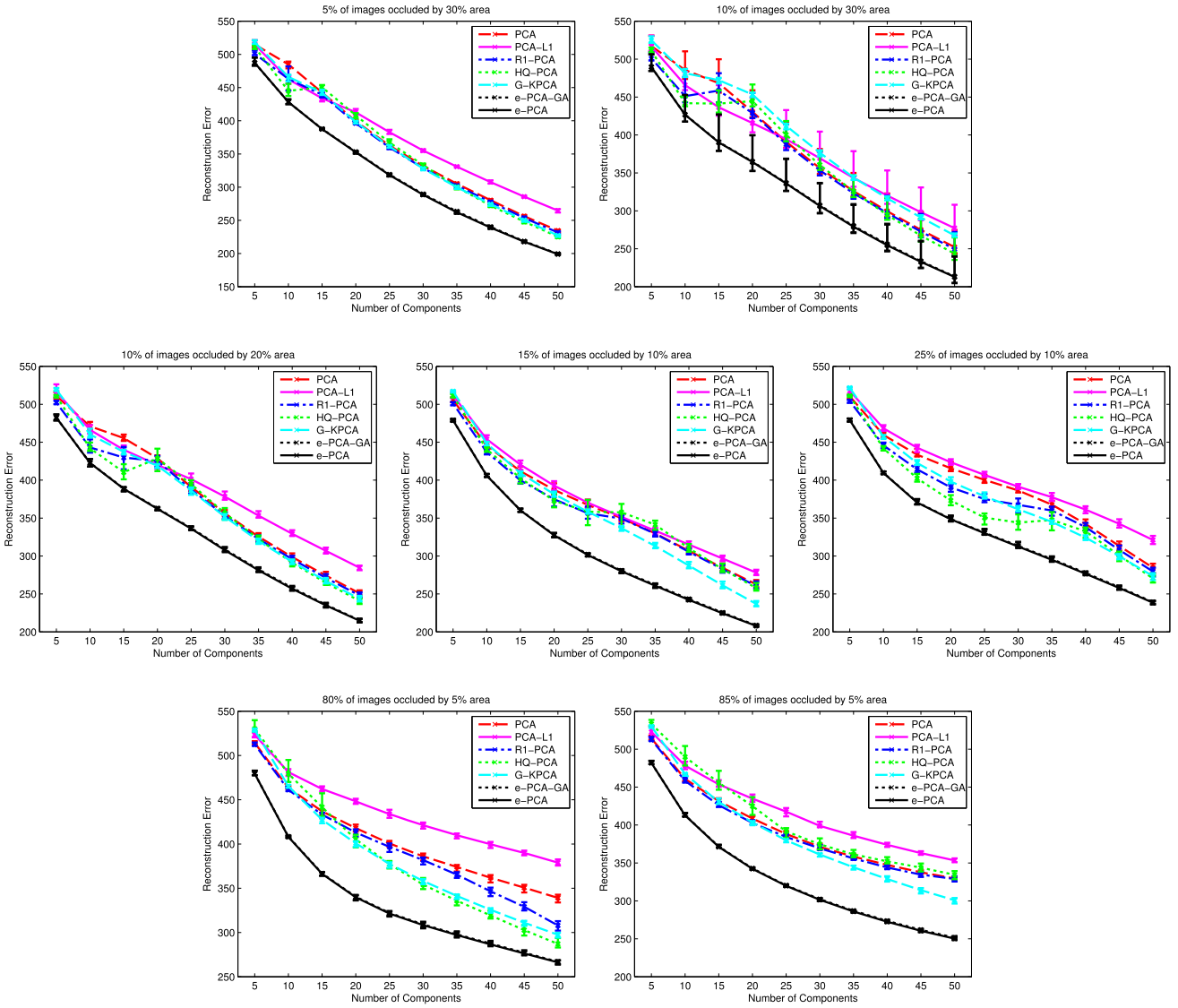

Fig. 6 Reconstruction error with different rates of occluded images. Here, the mean value over 10 executions with different random patches is shown. Variance is indicated by *error bars*

online learning is advantageous to fixed appearance models learned offline (Babenko et al. 2011; Ross et al. 2008; Mei and Ling 2009). We evaluate the tracking performance of Euler-PCA based on precision and accuracy and then compare it to four other state-of-the-art holistic trackers.

4.2.1 Framework

We combine the appearance model learned by the incremental version of Euler-PCA with a motion affine transformation and a particle filter, in a similar fashion to Ross et al. (2008) and Chin and Suter (2007). In general, a particle filter calculates the posterior of a system's states based on a transition model and an observation model. In our tracking framework, the transition model is described as a Gaussian Mixture Model around an approximation of the state posterior distribution of the previous time step,

$$p\left(A_t^i | A_{t-1}^{1:P}\right) = \sum_{i=1}^{P} w_{t-1}^i \mathcal{N}\left(A_t; A_{t-1}^i, \boldsymbol{\Xi}\right) \tag{24}$$

where A_t^i is the affine transformation of particle i at time t, $A_{t-1}^{1:P}$ is the set of P transformations of the previous time step, whose weights are denoted by $w_{t-1}^{1:P}$, and $\boldsymbol{\Xi}$ is an independent covariance matrix, which represents the variance in horizontal and vertical displacement, rotation, scale, ratio and shew. In the first phase, P particles are drawn from (24). In the second phase, the observation model is applied to estimate the weighting for the next iteration (the weights are normalized to ensure $\sum_{i=1}^{P} w_t^i = 1$). Furthermore, the most

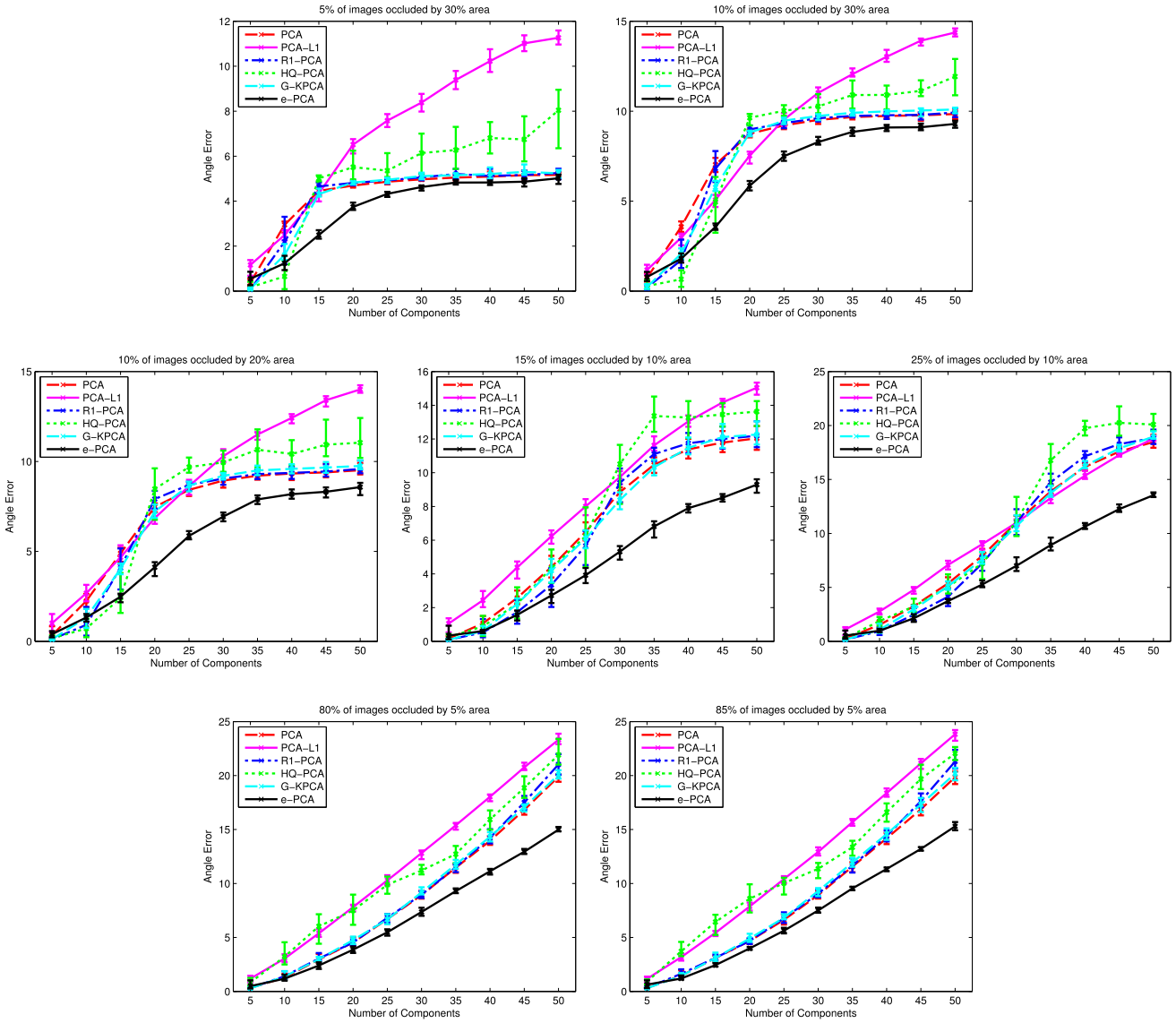

Fig. 7 Angular error with different rates of occluded images. Here, the mean value over 10 executions with different random patches is shown. Variance is indicated by *error bars*

probable sample is selected as the state A_t^{best} at time t. Thus, the estimation of the posterior distribution is an incremental process and utilizes a hidden Markov model which only relies on the previous time step.

Our observation model computes the probability of a sample being generated by the learned eigenspace in the appearance model. We assume that the probability of the observation, given the tracking parameters at t, is analogous to an exponential as

$$p\left(\phi\left(\mathbf{y}_t^i\right)|\mathbf{A}_t^i\right) \propto e^{-\gamma\|\phi(\mathbf{y}_t^i)-\mathbf{B}\mathbf{B}^H\phi(\mathbf{y}_t^i)\|_F^2} = e^{-\gamma\|\mathbf{z}_t^i-\mathbf{B}\mathbf{B}^H\mathbf{z}_t^i\|_F^2} \tag{25}$$

where $\mathbf{y}_t^i$ is the observation vector at time t of location $\mathbf{A}_t^i$, $\mathbf{z}_t^i$ is its mapping from (8) and γ is the parameter that controls the spread. Algorithm 4 describes the proposed visual tracking framework, which we coin Euler Kernel Tracker (eT).

Fig. 8 Examples of the corruptions in the training data

4.2.2 Results

We present the performance evaluation results of the proposed Euler Kernel Tracker (eT). We compare the performance of our method with that of four other state-of-the-

Fig. 9 Reconstructions of PCA, PCA-L1, R1-PCA, HQ-PCA, G-KPCA, *e*-PCA-GA and Euler-PCA (top to bottom) after learning with 80 % images occluded by an area of 5 % (20 components). The *last row* shows the corrupted training data. The uncorrupted images are shown in Fig. 3

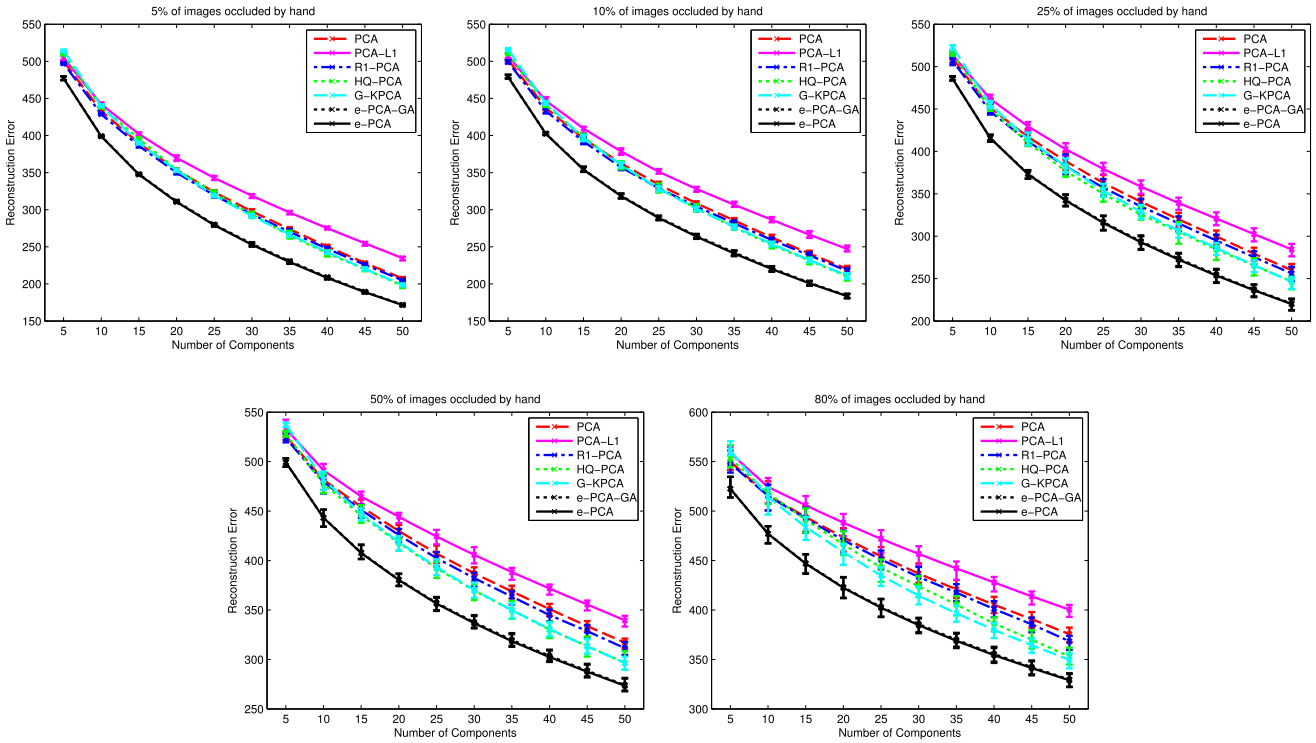

Fig. 10 Reconstruction error with different rates of hand occluded images. Here, the mean value over 10 executions is shown. Variance is indicated by *error bars*

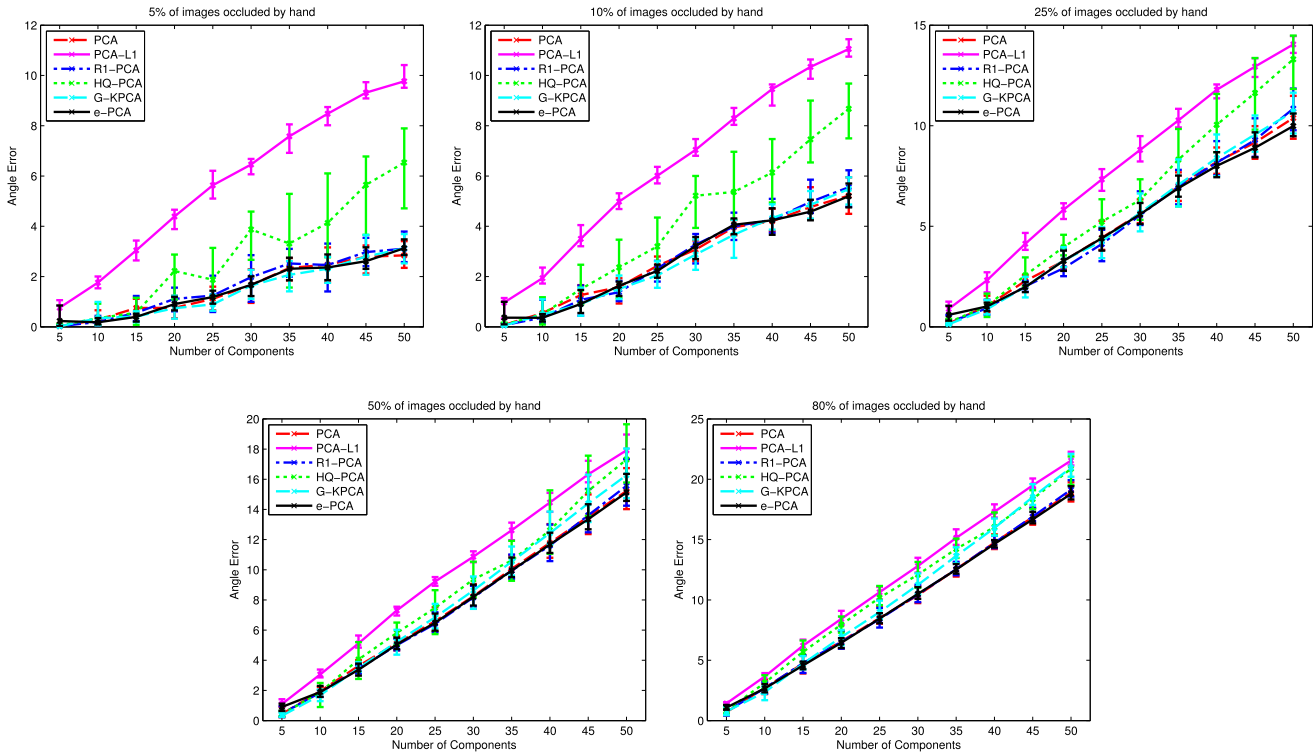

Fig. 11 Angular error with different rates of hand occluded images. Here, the mean value over 10 executions is shown. Variance is indicated by *error bars*

Fig. 12 Reconstructions of PCA, PCA-L1, R1-PCA, HQ-PCA, G-KPCA, *e*-PCA-GA and Euler-PCA (top to bottom) after learning with 50 % hand occluded images (20 components). The *bottom row* shows the training data. The uncorrupted images are shown in Fig. 3

Algorithm 4 EULER KERNEL TRACKER AT TIME t

Input: The previous eigenspace $\mathbf{B}_{t-1}$, $\mathbf{\Sigma}_{t-1}$, locations $\mathbf{A}_{t-1}^{1:P}$, weights $w_{t-1}^{1:P}$, image frame $\mathbf{I}_t \in [0, 1]$ and α.

1: Draw P particles $\mathbf{A}_t^{1:P}$ from $p(A_t^i|A_{t-1}^{1:P})$ as in (24).
2: Take all image patches from $\mathbf{I}_t$ which correspond to particles $\mathbf{A}_t^{1:P}$ and order their values lexicographically to form vectors $\mathbf{y}_t^{1:P}$.
3: Form the mapping $\mathbf{z}_t^{1:P}$ as in (8) and compute the probability of each particle $p(\mathbf{z}_t^i|\mathbf{A}_t^i)$ as in (25) and extract the weights $w_t^{1:P}$.
4: Choose $\mathbf{A}_t^{best}$ and $\mathbf{z}_t^{best}$ as the affine transform and features of the particle with the largest weight.
5: Using $\mathbf{z}_t^{best}$ update the subspace by applying **Algorithm 3** in a batch after a certain number of frames (5 in our implementation).

art tracking approaches: IVT[4] (Ross et al. 2008), IKPCA[5] (Chin and Suter 2007), the L1 tracker[6] (Mei and Ling 2009) and MIL tracker[7] (Babenko et al. 2011),

We evaluate the performance of all methods on 8 very popular video sequences (subsets of which are used in Babenko et al. (2009, 2011), Ross et al. (2008), Mei and Ling (2009), Comaniciu et al. (2003)), $V_i, i = 1, \ldots, 8$, with drastic changes of the target's appearance including pose variation, occlusions and non-uniform illumination.[8] Qualitative results are illustrated in Fig. 13.

Video V_1 is provided along with 7 annotated points which indicate the ground truth. We also annotate 3–7 fiducial points for the remaining sequences (Liwicki et al. 2012). Our quantitative performance evaluation is based on the root mean square (RMS) error between the true and the estimated locations of these points (Ross et al. 2008). Similarly to (Babenko et al. 2011), we additionally present precision plots which visualize the quality of the tracking. Such graphs show the percentage of frames in which the target was tracked with an RMS error less than a certain threshold.

In our experiments, all trackers use an affine motion model with a fixed number of drawn particles (800 particles). We attempt to optimize the performance of all trackers using video-specific parameters. That is, for each tracker and video, we found the parameters which gave the best performance in terms of robustness (*i.e.* how many times the tracker went completely off) and accuracy (measured by the RMS error).

Apart from the L1 tracker (for which the resolution of the template increases geometrically the complexity) the tracking template was chosen to be of resolution of 32×32. All trackers were optimized with respect to (wrt) the variance of the Gaussian from which we sample the particles. Additionally to the variance of the Gaussian, which is common for all the systems, we optimize eT, IVT and IKPCA wrt the number of components and the spread γ. For eT the value for α *is fixed to* 1.9. IKPCA was also optimized wrt the variance of the Gaussian RBF function. Furthermore, we optimized L1 wrt the resolution of the templates (the tracking becomes impractical for particles larger than 20×20). For MIL we optimized wrt the parameters mentioned in Babenko et al. (2011) (*e.g.* the number of positives in each frame, the number that controls the sampling of negative examples, the learning rate for the weak classifiers).

For these versions of the trackers, Table 2 lists the mean RMS error for all sequences and the average frame rate of each tracker,[9] while Fig. 14 plots the RMS error as a function of the frame number. Figure 15 shows the accuracy in terms of precision plots. Qualitative tracking results for all methods are shown in Fig. 13. The videos are provided as part of the supplementary material.

In general, the robustness of eT is similar to IVT, although, eT performs the best in terms of precision for most videos. MIL and L1 are more robust and track the target in V_5 successfully. However, particularly visible in the results for V_8, L1 is not precise for outliers caused by motion blur. MIL is based on a bag of features approach, and consequently is inherently unprecise. IKPCA fails for all sequences. Our tracker performs very well particularly for V_4 in which the target undergoes many prolonged partial occlusions. The e-PCA's robustness successfully suppresses these outliers for this video sequence. In terms of efficiency, IVT and e-PCA operate in the highest frame rate, while all other methods operate in less than one frame per second.

4.3 Background Modeling

Background modeling algorithms aim to estimate the background of a scene from a video sequence usually captured with a static camera. This problem can be naturally tackled using PCA (Oliver et al. 2000): the frames of the video are used to estimate a low dimensional subspace. Then the

[4]The Matlab implementation is publicly available at http://www.cs.toronto.edu/~dross/ivt/.

[5]The Matlab implementation of the IKPCA was kindly provided by the authors of the paper.

[6]The implementation is publicly available at http://www.ist.temple.edu/~hbling/code_data.htm.

[7]The implementation (only for translation motion model) is publicly available at http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml, we carefully modified it in order to support an affine motion model in a particle filter framework.

[8]Videos V_4 and V_5 are available at http://vision.ucsd.edu/~bbabenko/project_miltrack.shtml and the remaining videos are published at http://www.cs.toronto.edu/~dross/ivt/.

[9]MATLAB implementations on a desktop computer with Intel Core i7 870 at 2.93 GHz and 8 GB RAM.

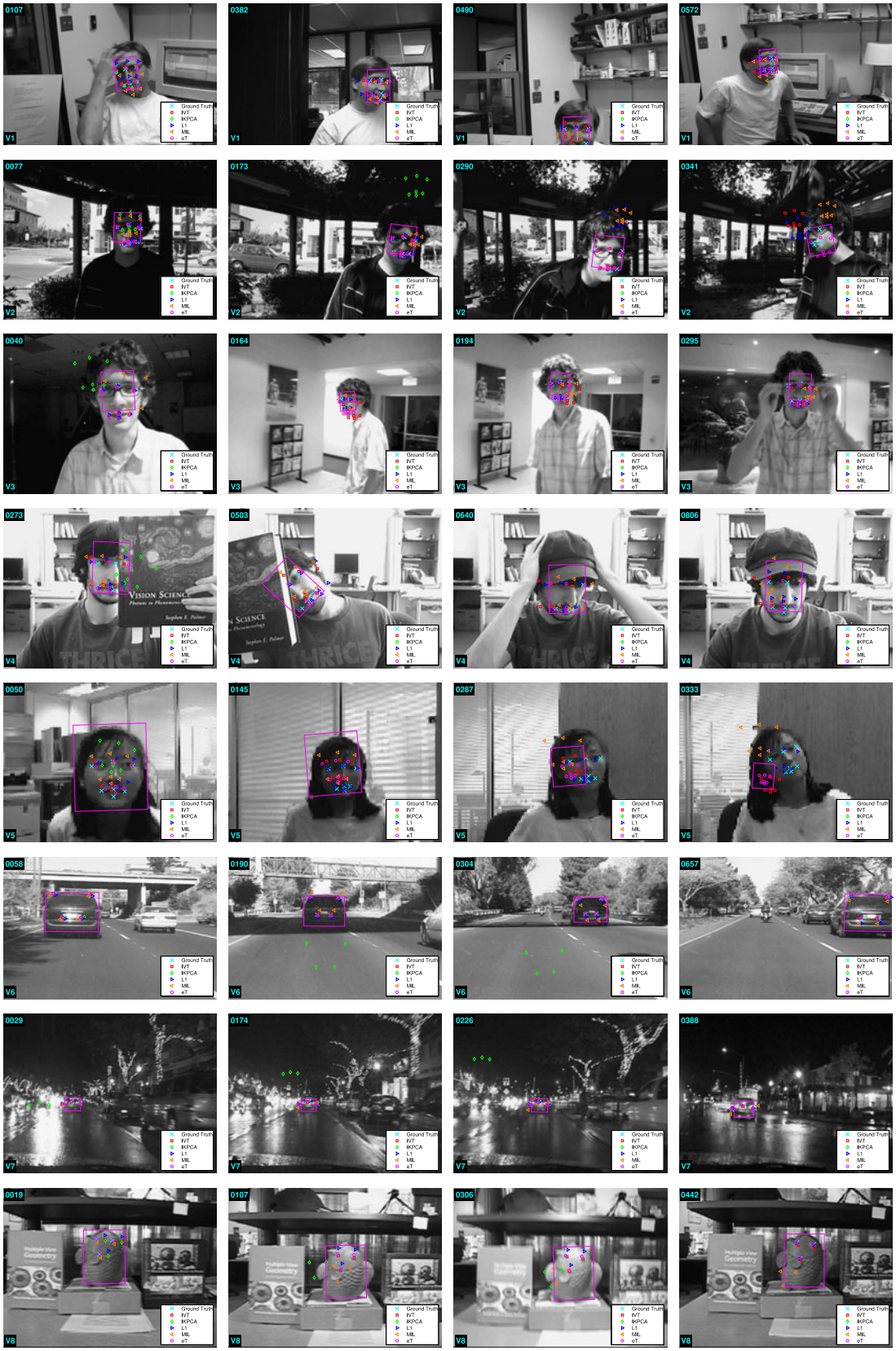

Fig. 13 Qualitative tracking results for videos V_1 to V_8

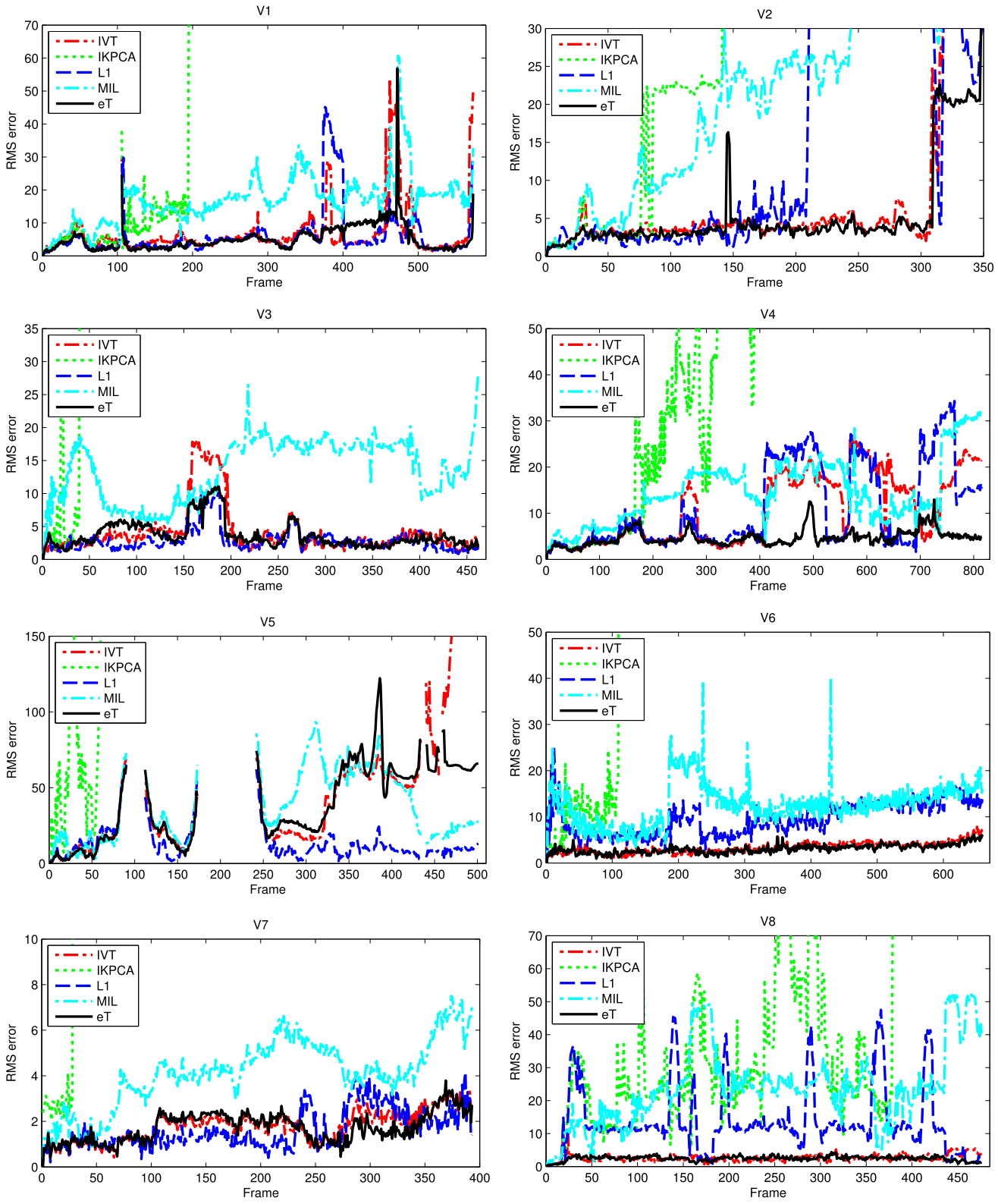

Fig. 14 Mean root square error achieved by all tested trackers as a function of the frame number for each video sequence

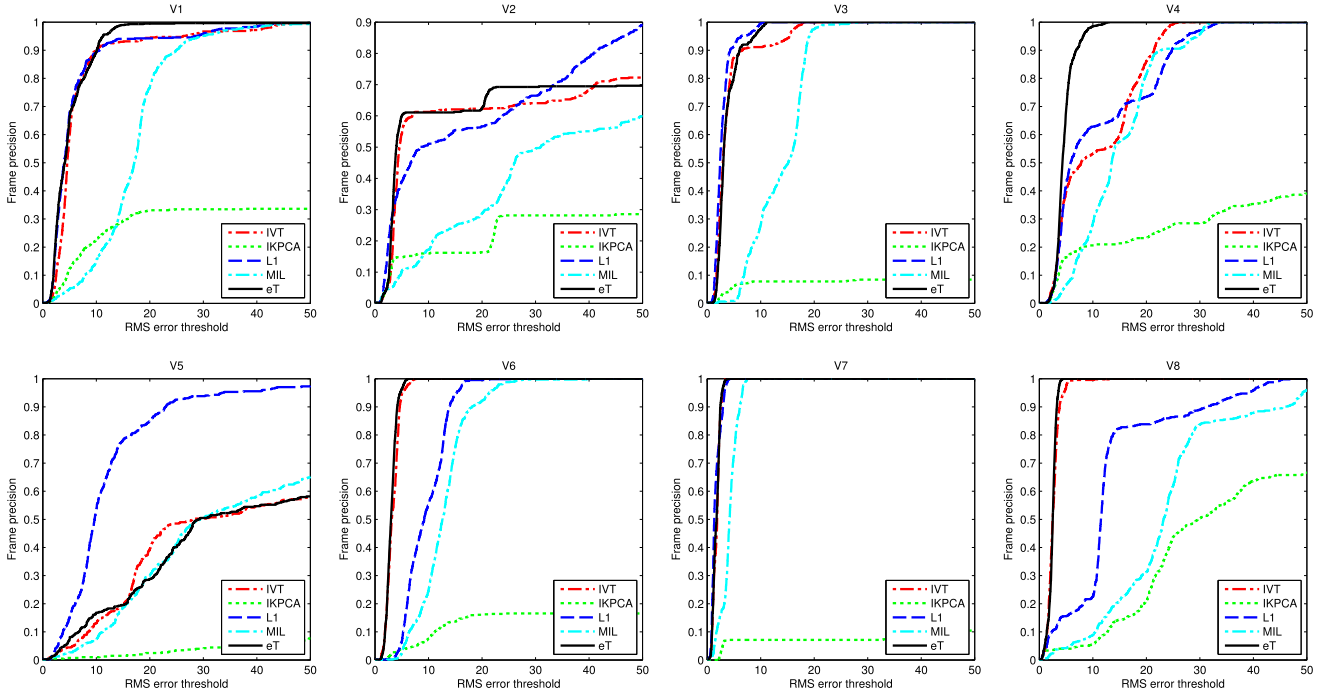

Fig. 15 Tracking precision for each video sequence

Table 2 Mean RMS error for general tracking, and tracking rate. "(lost)" indicates sequences in which the tracker clearly does not follow the target throughout

	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	Fames/second
IVT	6.82	(lost)	4.07	10.79	(lost)	3.31	1.78	2.62	**3.157**
IKPCA	(lost)	(lost)	(lost)	(lost)	(lost)	(lost)	(lost)	(lost)	0.832
L1	6.17	(lost)	**2.87**	11.10	**12.68**	9.53	1.62	13.58	0.076
MIL	16.95	(lost)	13.61	14.62	37.56	12.73	4.14	23.87	0.129
***e*T**	**5.14**	(lost)	3.68	**4.68**	(lost)	**3.04**	**1.73**	**2.44**	2.935

Table 3 Maximum similarity of Fig. 16

	Airport	Bar	Lobby	Curtain	Escalator	Fountain	Mall	Campus	Water
PCA	0.540	0.503	0.600	0.686	0.442	0.508	0.545	0.286	0.767
PCA-L1	0.540	0.504	0.604	0.671	0.442	0.548	0.540	0.286	0.767
R1-PCA	0.474	0.499	0.607	0.727	0.428	**0.563**	0.551	0.294	0.503
HQ-PCA	0.486	0.498	**0.615**	**0.755**	0.422	0.558	0.544	0.292	**0.776**
***e*-PCA**	**0.584**	**0.533**	0.609	0.747	**0.479**	0.534	**0.563**	**0.304**	0.774

Table 4 Execution time required to compute the appearance model for the last frame of each video sequence (5 components)

	Airport	Bar	Lobby	Curtain	Escalator	Fountain	Mall	Campus	Water
PCA	0.003 s	0.002 s	0.003 s	0.002 s	0.002 s	0.002 s	0.002 s	0.010 s	0.003 s
PCA-L1	12.4 s	9.5 s	5.1 s	6.3 s	21.7 s	1.2 s	3.3 s	9.7 s	1.4 s
R1-PCA	123.0 s	120.0 s	44.7 s	124.9 s	233.0 s	12.4 s	35.9 s	182.8 s	8.1 s
HQ-PCA	2663.4 s	930.8 s	365.8 s	3106.8 s	2536.7 s	36.7 s	91.0 s	282.3 s	71.8 s
***e*-PCA**	0.007 s	0.006 s	0.006 s	0.006 s	0.006 s	0.006 s	0.006 s	0.026 s	0.006 s

background corresponding to each of the video frames is obtained by reconstructing the frame from this subspace. Once the background estimate is obtained, the foreground objects can be segmented typically by subtraction and thresholding.

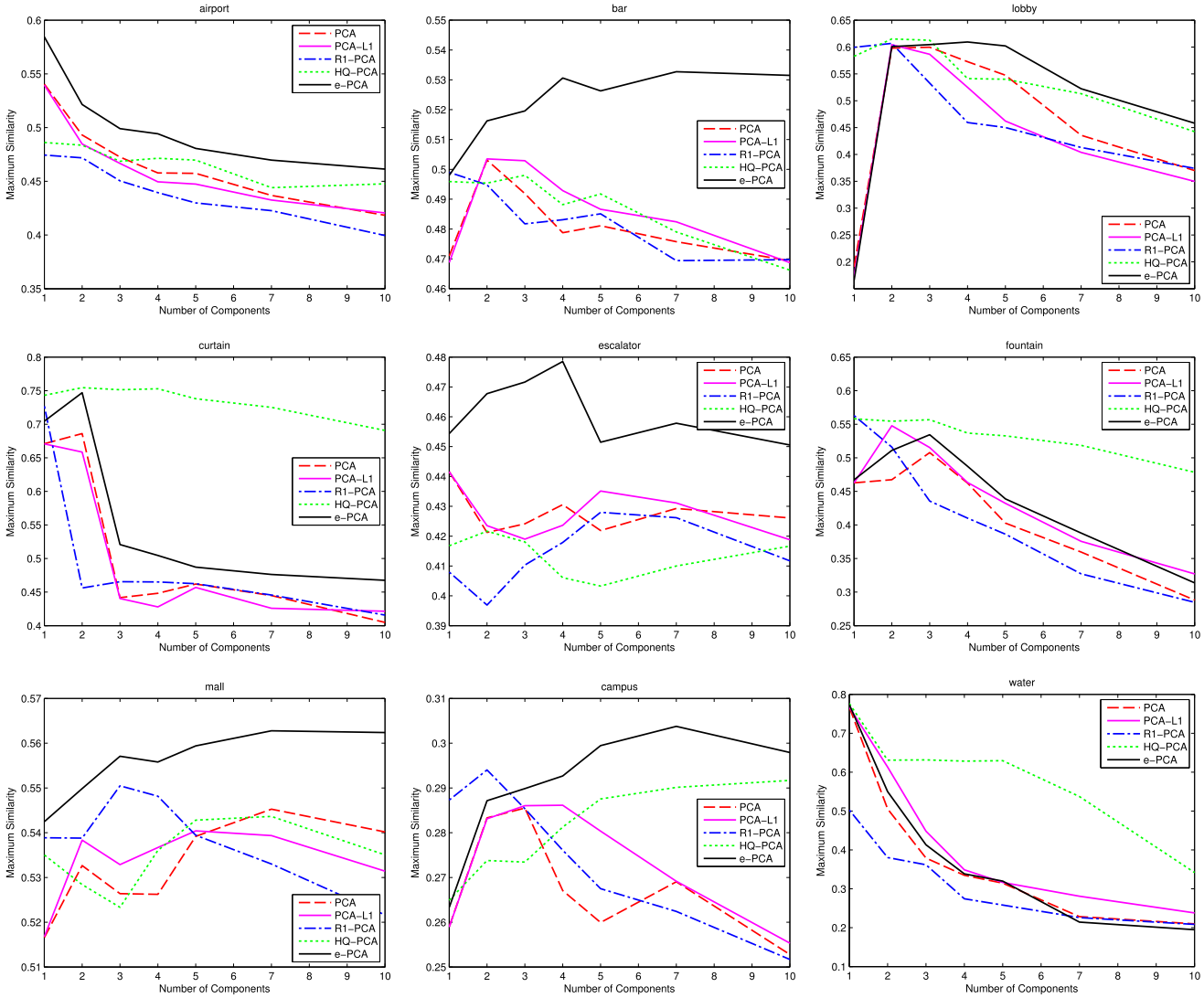

Fig. 16 Similarity with changing number of components

For our evaluation, we used the popular data set from Li et al. (2004). The set consists of 9 videos including illumination changes, indoor/outdoor environments as well as dynamic background changes. The ground truth for foreground/background pixels of 20 randomly selected frames for each video is also provided Li et al. (2004). Standard PCA, PCA-L1, R1-PCA and HQ-PCA are used for comparison. We present quantitatively and qualitatively results. For the former case, we use the similarity measure (Maddalena and Petrosino 2008)

$$Similarity = \frac{tp}{tp + fp + fn} \tag{26}$$

where tp, fp and fn are the numbers of correctly labeled foreground, falsely labeled background and falsely labeled foreground pixels respectively. The setup for PCA, PCA-L1, R1-PCA, HQ-PCA and e-PCA is similar to Sect. 4.1. Furthermore, PCA and Euler-PCA is updated incrementally for each frame during learning.

We used the complete set of preceding frames to train the models (*e.g.* for frame 100, the preceding 99 frames are used for the appearance model), and for each video, we evaluate the similarity for the frames in which the ground truth is provided. The mean similarity, as a function of the number of components, is plotted in Fig. 16. The best similarity value for each method and video is summarized in Table 3, while Fig. 17 shows the performance qualitatively. In general, e-PCA performs the best in 5 out of 9 sequences, and the second best for 3. The results of the other methods vary for each sequence. The videos are provided in the supplementary material.

In Table 4 we highlight the computation time of the appearance model for the final frame of each video sequence. Our tests were conducted in MATLAB (64 bit) on a desktop

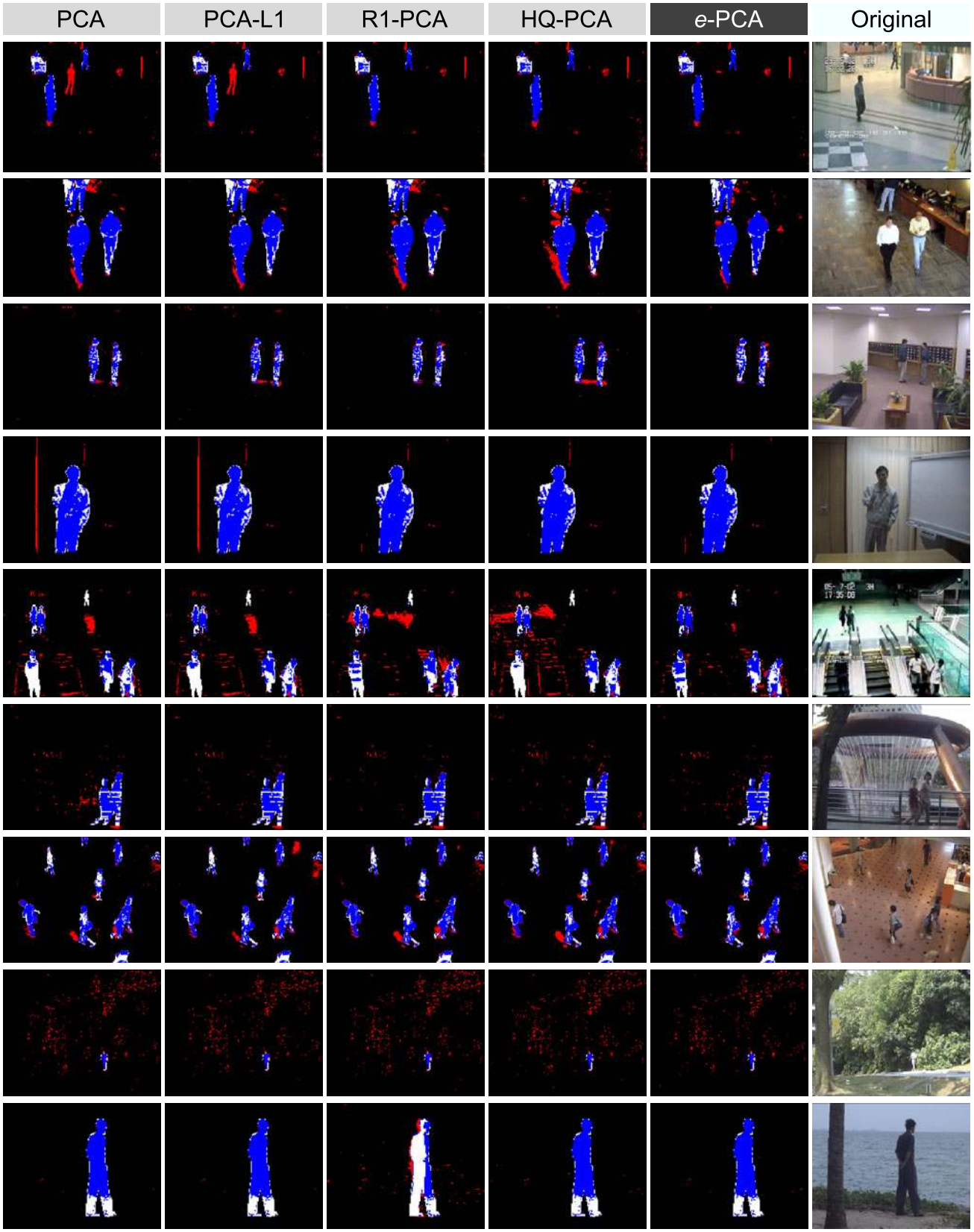

Fig. 17 Examples of background modeling for each video and each method. In the results, *black* indicates correctly predicted background, *blue* indicates correctly predicted foreground, *red* indicates misclassified background and white indicates misclassified foreground

computer with an Intel core i7 870 processor at 2.93 GHz and 8 GB RAM. PCA and Euler-PCA can be updated incrementally, making their running time less than a second for all sequences. In contrast, the other methods require a recalculation of the complete appearance model for each frame. Consequently, these methods are much slower.

5 Conclusion

We introduce a fast, direct and robust approach to PCA. The proposed Euler-PCA allows for fast incremental computation and retains the favorable properties of standard ℓ_2-norm PCA, while suppressing outliers. Our experiments show that Euler-PCA achieves promising results for the applications of face reconstruction, object tracking and background modeling. In future work we intend to introduce e-PCA to a range of further applications in human computer interaction, computer vision and pattern recognition.

Acknowledgements The research presented in this paper is supported in part by the European Research Council (ERC) under the ERC Starting Grant Agreement ERC-2007- StG-203143 (MAHNOB). The work of S. Liwicki is supported by the Engineering and Physical Science Research Council DTA Studentship. The work of G. Tzimiropoulos is currently supported in part by the European Community's 7th Framework Programme FP7/2007-2013 under Grant Agreement 288235 (FROG).

Appendix A: Proof of Theorem 1

Proof Given $\mathbf{A} = \boldsymbol{\Phi}\boldsymbol{\Phi}^H$ and $\mathbf{B} = \boldsymbol{\Phi}^H\boldsymbol{\Phi}$ their eigenspaces is provided by $\mathbf{A} = \mathbf{U}_A\boldsymbol{\Lambda}_A\mathbf{U}_A^H$ and $\mathbf{B} = \mathbf{U}_B\boldsymbol{\Lambda}_B\mathbf{U}_B^H$. Furthermore, $\mathbf{U}_A^H\mathbf{U}_A = \mathbf{U}_B^H\mathbf{U}_B = \mathbf{I}$. Let us define matrix $\mathbf{M} = \boldsymbol{\Phi}\mathbf{U}_B\boldsymbol{\Lambda}_B^{-\frac{1}{2}}$. We get

$$\begin{aligned}\mathbf{M}^H\mathbf{A}\mathbf{M} &= \boldsymbol{\Lambda}_B^{-\frac{1}{2}}\mathbf{U}_B^H\boldsymbol{\Phi}^H\boldsymbol{\Phi}\boldsymbol{\Phi}^H\boldsymbol{\Phi}\mathbf{U}_B\boldsymbol{\Lambda}_B^{-\frac{1}{2}}\\ &= \boldsymbol{\Lambda}_B^{-\frac{1}{2}}\mathbf{U}_B^H\mathbf{B}\mathbf{B}\mathbf{U}_B\boldsymbol{\Lambda}_B^{-\frac{1}{2}}\\ &= \boldsymbol{\Lambda}_B^{-\frac{1}{2}}\mathbf{U}_B^H\mathbf{U}_B\boldsymbol{\Lambda}_B\mathbf{U}_B^H\mathbf{U}_B\boldsymbol{\Lambda}_B\mathbf{U}_B^H\mathbf{U}_B\boldsymbol{\Lambda}_B^{-\frac{1}{2}}\\ &= \boldsymbol{\Lambda}_B^{-\frac{1}{2}}\boldsymbol{\Lambda}_B\boldsymbol{\Lambda}_B\boldsymbol{\Lambda}_B^{-\frac{1}{2}}\\ &= \boldsymbol{\Lambda}_B. \end{aligned} \tag{27}$$

Therefore, $\boldsymbol{\Lambda}_A = \boldsymbol{\Lambda}_B$ and $\mathbf{U}_A = \mathbf{M}$ for non-zero eigenvalues. □

Appendix B: Proof that $\|\frac{1}{\sqrt{2}}e^{i\angle\mathbf{b}} - \mathbf{b}\|_F^2 = \|\frac{1}{\sqrt{2}} - \mathtt{R}(\mathbf{b})\|_F^2$

$$\begin{aligned}\left\|\frac{1}{\sqrt{2}}e^{i\angle\mathbf{b}} - \mathbf{b}\right\|_F^2 &= \sum_{c=1}^{p}\left(\frac{1}{\sqrt{2}}e^{i\angle\mathbf{b}(c)} - \mathbf{b}(c)\right)^2\\ &= \sum_{c=1}^{p}\left(\frac{1}{\sqrt{2}}e^{i\angle\mathbf{b}(c)} - \mathtt{R}\big(\mathbf{b}(c)\big)e^{i\angle\mathbf{b}(c)}\right)^2\\ &= \sum_{c=1}^{p}\left(\frac{1}{\sqrt{2}} - \mathtt{R}\big(\mathbf{b}(c)\big)\right)^2\\ &= \left\|\frac{1}{\sqrt{2}}\mathbf{1} - \mathtt{R}(\mathbf{b})\right\|_F^2, \end{aligned} \tag{28}$$

where $\mathtt{R}(\mathbf{b}) = [\sqrt{\mathtt{Re}(\mathbf{b}(c))^2 + \mathtt{Im}(\mathbf{b}(c))^2}]$ is a vector with the magnitude of the elements of $\mathbf{b}$ and $\mathbf{1}$ is a vector of ones. □

References

Avidan, S. (2004). Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1064–1072. doi:10.1109/TPAMI.2004.53.

Babenko, B., Yang, M., & Belongie, S. (2009). Visual tracking with online multiple instance learning. In *CVPR'09* (pp. 983–990).

Babenko, B., Yang, M., & Belongie, S. (2011). Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. doi:10.1109/TPAMI.2010.226.

Candés, E., Li, X., Ma, Y., & Wright, J. (2009). Robust principal component analysis? Available at: http://arxiv.org/abs/0912.3599v1.

Chin, T. J., & Suter, D. (2007). Incremental kernel principal component analysis. *IEEE Transactions on Image Processing*, 1662–1674. doi:10.1109/TIP.2007.896668.

Chin, T., Schindler, K., & Suter, D. (2006). Incremental kernel SVD for face recognition with image sets. In *FG'06* (pp. 461–466).

Cohn, J., Zlochower, A., Lien, J., & Kanade, T. (1999). Automated face analysis by feature point tracking has high concurrent validity with manual FACS coding. *Psychophysiology*, 35–43. doi:10.1017/S0048577299971184.

Collins, R., Lipton, J., Fujiyoshi, H., & Kanade, T. (2001). Algorithms for cooperative multisensor surveillance. In *The IEEE* (p. 89). doi:10.1109/5.959341.

Comaniciu, D., Ramesh, V., & Meer, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 564–577. doi:10.1109/TPAMI.2003.1195991.

de la Torre, F., & Black, M. (2003). A framework for robust subspace learning. *International Journal of Computer Vision*, 117–142. doi:10.1023/A:1023709501986.

Ding, D., Zhou, D., He, X., & Zha, H. (2006). R1-PCA: rotational invariant L1-norm principal component analysis for robust subspace factorization. In *ACM* (pp. 281–288). doi:10.1145/1143844.1143880.

Fitch, A., Kadyrov, A., Christmas, W., & Kittler, J. (2005). Fast robust correlation. *IEEE Transactions on Image Processing*, 1063–1073. doi:10.1109/TIP.2005.849767.

Fraundorfer, F., Engels, C., & Nistér, D. (2007). Topological mapping, localization and navigation using image collections. In *Intell. robots and systems* (pp. 3872–3877).

Gunawan, H., Neswan, O., & Budhi, W. (2005). A formula for angles between subspaces of inner product spaces. *Contributions to Algebra and Geometry*, *46*(2), 311–320.

Gunes, H., & Pantic, M. (2010). Automatic, dimensional and continuous emotion recognition. *International Journal of Synthetic Emotion*, 68–99. doi:10.4018/jse.2010101605.

Handmann, U., Kalinke, T., & Tzomakas, C. (1998). Computer vision for driver assistance systems. *Proc. SPIE*, 136–147. doi:10.1117/12.317463.

Haritaoglu, I., Harwood, D., & Davis, L. (2000). W4: real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 809–830. doi:10.1109/34.868683.

Hashima, M., Hasegawa, F., Kanda, S., Maruyama, T., & Uchiyama, T. (1997). Localization and obstacle detection for a robot for carrying food trays. In *Intell. robots and systems* (pp. 345–351).

He, R., Hu, B., Zheng, W., & Kong, X. (2011). Robust principal component analysis based on maximum correntropy criterion. *IEEE Transactions on Image Processing*, 1485–1494. doi:10.1109/TIP.2010.2103949.

Honeine, P., & Richard, C. (2011). Preimage problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, *28*(2), 77–88.

Hsieh, J., Yu, S., Chen, Y., & Hu, W. (2006). Automatic traffic surveillance system for vehicle tracking and classification. *IEEE Transactions on Intelligent Transportation Systems*, 175–187. doi:10.1109/TITS.2006.874722.

Jolliffe, T. (2002). *Principal component analysis* (2nd edn.). Berlin: Springer.

Kamijo, S., Matsushita, Y., Ikeuchi, K., & Sakauchi, M. (2000). Traffic monitoring and accident detection at intersections. *IEEE Transactions on Intelligent Transportation Systems*, 108–118. doi:10.1109/6979.880968.

Ke, Q., & Kanade, T. (2003). *Robust subspace computation using L1 norm* (Tech. Rep. CMU-CS-03-172). Computer Science Department, Carnegie Mellon University.

Ke, Q., & Kanade, T. (2005). Robust L1 norm factorization in the presence of outliers and missing data by alternative convex programming. In *CVPR'05* (pp. 739–746).

Krzanowski, W. (1979). Between-groups comparison of principal components. *Journal of the American Statistical Association*, 703–707. doi:10.1080/01621459.1979.10481674.

Kwak, N. (2008). Principal component analysis based on L1-norm maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1672–1680. doi:10.1109/TPAMI.2008.114.

Kwok, J., & Tsang, I. (2004). The pre-image problem in kernel methods. *IEEE Transactions on Neural Networks*, *15*(6), 1517–1525.

Levy, A., & Lindenbaum, M. (2000). Sequential Karhunen-Loeve basis extraction and its application to images. *IEEE Transactions on Image Processing*, 1371–1374. doi:10.1109/83.855432.

Li, Y. (2004). On incremental and robust subspace learning. *Pattern Recognition*, 1509–1518. doi:10.1016/j.patcog.2003.11.010.

Li, L., Huang, W., Gu, I., & Tian, Q. (2004). Statistical modeling of complex backgrounds for foreground object detection. *IEEE Transactions on Image Processing*, *13*(11), 1459–1472.

Liu, W., Pokharel, P., & Principe, J. (2007). Correntropy: properties and applications in non-Gaussian signal processing. *IEEE Transactions on Signal Processing*, 5286–5298. doi:10.1109/TSP.2007.896065.

Liwicki, S., & Everingham, M. (2009). Automatic recognition of fingerspelled words in British sign language. In *CVPR4HB'09, in conj. with CVPR'09* (pp. 50–57).

Liwicki, S., et al. (2012). doi:10.1109/TNNLS.2012.2208654.

Luo, Y., Wu, T., & Hwang, J. (2003). Object-based analysis and interpretation of human motion in sports video sequences by dynamic Bayesian networks. *Computer Vision and Image Understanding*, 196–216. doi:10.1016/j.cviu.2003.08.001.

Maddalena, L., & Petrosino, A. (2008). A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Transactions on Image Processing*, 1168–1177. doi:10.1109/TIP.2008.924285.

Martinez, A., & Benavente, R. (1998). *The AR face database* (Tech. Rep. #24). The Ohio State University.

Mei, X., & Ling, H. (2009). Robust visual tracking using L1 minimization. In *ICCV'09*.

Mika, S., Schölkopf, B., Smola, A., Müller, K., Scholz, M., & Rätsch, G. (1999). Kernel pca and de-noising in feature spaces. *Advances in Neural Information Processing Systems*, *11*(1), 536–542.

Nguyen, M., & de la Torre, F. (2009). Robust kernel principal component analysis. In *Advances in NIPS* (pp. 1185–1192).

Oliver, N., Rosario, B., & Pentland, A. (2000). A Bayesian computer vision system for modeling human interactions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 831–843. doi:10.1109/34.868684.

Paulsen, V. (2009). An introduction to the theory of reproducing kernel Hilbert spaces. Available at: http://www.math.uh.edu/~vern/rkhs.pdf.

Ross, D., Lim, J., Lin, R., & Yang, M. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 125–141. doi:10.1007/s11263-007-0075-7.

Tzimiropoulos, G. (2010). *IEEE Transactions on Pattern Analysis and Machine Intelligence*. doi:10.1109/TPAMI.2010.107.

Tzimiropoulos, G. (2012). *IEEE Transactions on Pattern Analysis and Machine Intelligence*. doi:10.1109/TPAMI.2012.40.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 71–86. doi:10.1162/jocn.1991.3.1.71.

Wren, C., Azarbayejani, A., Darrel, T., & Pentland, A. (1997). Pfinder: real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 780–785. doi:10.1109/34.598236.