A Framework for Robust Subspace Learning

Fernando De la Torre
Department of Communications and Signal Theory, La Salle School of Engineering,
Universitat Ramon LLull, Barcelona 08022, Spain

Phone: +34 93 290 2440, FAX: +34 93 290 2416
Email: ftorre@salleURL.edu

Michael J. Black
Department of Computer Science, Brown University, Box 1910,
Providence, RI 02912, USA

Phone: +1 401-863-7637, FAX: +1 401-863-7657
Email: black@cs.brown.edu

July 25, 2002

Abstract

Many computer vision, signal processing and statistical problems can be posed as problems of learning low dimensional linear or multi-linear models. These models have been widely used for the representation of shape, appearance, motion, etc, in computer vision applications. Methods for learning linear models can be seen as a special case of subspace fitting. One drawback of previous learning methods is that they are based on least squares estimation techniques and hence fail to account for "outliers" which are common in realistic training sets. We review previous approaches for making linear learning methods robust to outliers and present a new method that uses an *intra-sample* outlier process to account for pixel outliers. We develop the theory of *Robust Subspace Learning* (RSL) for linear models within a continuous optimization framework based on robust M-estimation. The framework applies to a variety of linear learning problems in computer vision including eigen-analysis and structure from motion. Several synthetic and natural examples are used to develop and illustrate the theory and applications of robust subspace learning in computer vision.

Keywords: Principal Component Analysis, Singular Value Decomposition, Learning, Robust statistics, Subspace Methods, Structure from Motion.

Submitted to: IJCV Special Issue on Vision at Brown.

1 Introduction

Automated learning of low-dimensional linear or multi-linear models from training data has become a standard paradigm in computer vision. A variety of linear learning models and techniques such as Principal Component Analysis (PCA) [20, 37, 38, 44, 67], Factor Analysis (FA) [22, 44], Autoregressive analysis (AR) [9], and Singular Value Decomposition (SVD) [28], have been widely used for the representation of high dimensional data such as appearance, shape, motion, temporal dynamics, etc. These approaches differ in their noise assumptions, the use of prior information, and the underlying statistical models, but all of them are directly or indirectly related to linear or bilinear regression. Learning linear models such as these can be posed as a problem of alternated least squares (ALS) estimation which is sometimes referred to as criss-cross regression [25]. In this paper we develop a robust formulation of this estimation processes which can be exploited to improve the robustness of linear learning methods to statistical outliers.

In particular, PCA is a popular technique for parameterizing shape, appearance, and motion [8, 15, 48, 50, 67]. Learned PCA representations have proven useful for solving problems such as face and object recognition, tracking, detection, and background modeling [5, 15, 48, 50, 52]. Typically, the training data for PCA is pre-processed in some way (e.g. faces are aligned [48]) or is generated by some other vision algorithm (e.g. optical flow is computed from training data [8]). As automated learning methods are applied to more realistic problems, and the amount of training data increases, it becomes impractical to manually verify that all the data is "good". In general, training data may contain undesirable artifacts due to occlusion (e.g. a hand in front of a face), illumination (e.g. specular reflections), image noise (e.g. from scanning archival data), or errors from the underlying data generation method (e.g. incorrect optical flow vectors). We view these artifacts as statistical "outliers" [56] and develop a theory of Robust Subspace Learning (RSL) for PCA that can be used to construct low-dimensional linear-subspace representations from noisy data. PCA provides a simple domain in which to motivate, develop, and illustrate the approach. We then show how this general framework can be extended to a variety of linear, or multi-linear, learning problems.

It is commonly known that traditional PCA constructs the rank k subspace approximation to zero-mean training data that is optimal in a least-squares sense [20, 21, 28, 38]. It is also com-

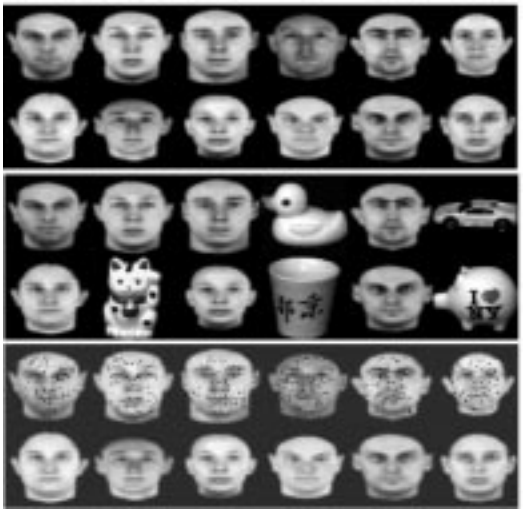
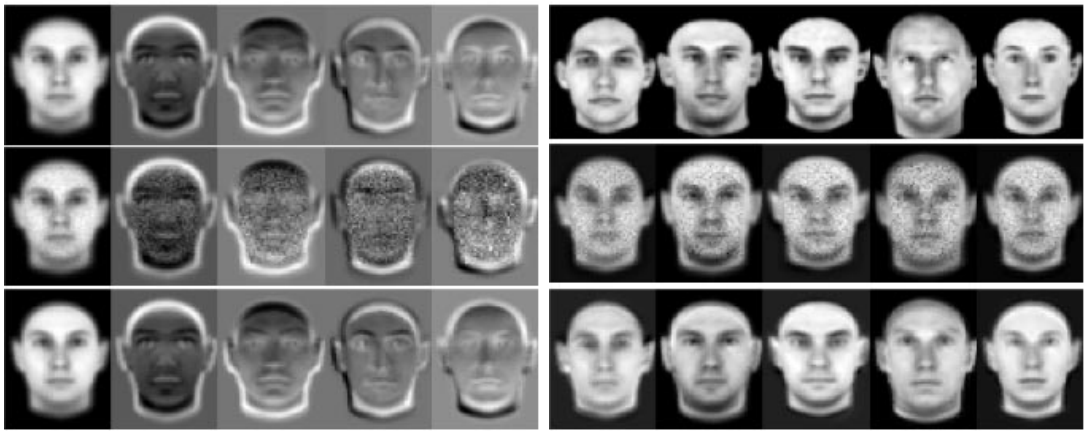


Figure 1: Illustrative training set and different types of outliers. *Top:* A few images from the original training set of 100 images. *Middle:* Training set with *sample outliers*. *Bottom:* Training set with *intra-sample outliers*.

monly known that least-squares techniques are not robust in the sense that outlying measurements can arbitrarily skew the solution from the desired solution [32, 35]. In the vision community, previous attempts to make PCA robust [70] have treated entire data samples (i.e. images) as outliers. This approach is appropriate when entire data samples are contaminated as illustrated in Figure 1 (*middle*). As argued above, the more common case in computer vision applications involves *intra-sample* outliers which affect some, but not all, of the pixels in a data sample (Figure 1 (*bottom*)).

Figure 2 presents a simple example to illustrate the effect of intra-sample outliers. The first row of Figure 2*a* shows the mean and the first four principal components for the 100 image training set of Figure 1 (*top*). The second row shows the bases recovered using PCA for the training set in Figure 1 (*bottom*) which contains intra-sample outliers. Notice that the outliers have affected all the basis images. By accounting for intra-sample outliers, the Robust Principal Component Analysis (RPCA) method described here constructs the linear basis shown in Figure 2 (*bottom*) in which the influence of outliers is reduced and the recovered bases are visually similar to those produced with traditional PCA on data without outliers.

Figure 2*b* shows the effect of outliers on the reconstruction of images using the learned linear subspaces. The first row shows noiseless images that were not present in the training set of faces. The middle row shows the reconstruction obtained by projecting each face onto the PCA basis



a *b*

Figure 2: *(a).* Effect of intra-sample outliers on learned basis images. *Top:* Standard PCA applied to noise-free data. *Middle:* Standard PCA applied to the training set corrupted with intra-sample outliers. *Bottom:* Robust PCA applied to corrupted training data. *(b).* Reconstruction results using subspaces constructed from noisy training data. *Top:* Original, noiseless, test images. *Middle:* Least-squares reconstruction of images with standard PCA basis (MSRE 19.35) . *Bottom:* Reconstructed images using RPCA basis (MSRE 16.54) .

images learned with the corrupted training data. This projection operation corresponds a least-squares estimate of the linear reconstruction coefficients and, hence, is influenced by the outlying data in the training set. The "mottled" appearance of the least squares method is not present when using the robust technique (*bottom*) and the Mean Squared Reconstruction Error (MSRE, defined below) is reduced.

In the following section we review previous work in the statistics, neural-networks, and vision communities that has addressed the robustness of subspace methods. In particular, we describe the method of Xu and Yuille [70] in detail and quantitatively compare it with our method and standard PCA. We show how linear (and multi-linear in general) methods can be modified by the introduction of an outlier process [6, 26] that can account for outliers at the pixel level. A robust M-estimation method is derived and details of the algorithm, its complexity, and its convergence properties are described. Like all M-estimation methods, the robust subspace learning (RSL) formulation has an inherent scale parameter that determines what is considered an outlier. We present a method for estimating this parameter from the data resulting in a fully automatic learning method. Synthetic experiments are used to illustrate how different robust approaches treat outliers and to quantitatively evaluate the method. Results on natural images show how the method can be used

to robustly learn a subspace of illumination variation for background modeling.

2 Previous work

A full review of linear learning methods and applications in computer vision is beyond the scope of this paper. For concreteness we focus on principal component analysis and then show how the robust methods generalize to other linear learning methods. For illustrative purposes and without loss of generality, we will use examples of learning models of images. The advantage of considering PCA for this task is that it is widely applicable and there has already been work in the vision community on improving its robustness.

Our formulation here is based on the techniques of robust M-estimation developed in the statistics community [32, 35]. The goal is to recover the solution (i.e. the learned model) that best fits the majority of the data and to detect and downweight "outlying" data. Loosely, the term "outlier" refers to data that does not conform to the assumed statistical model. A "robust" estimation method is one that can tolerate some percentage of outlying data without having the solution arbitrarily skewed. In computer vision applications, outliers are typically not "noise" in a traditional sense but rather are violations of highly simplified models of the the world; for example, the presence of specular reflections when one assumes Lambertian reflectance or the violation of the brightness constancy assumption at motion boundaries [4]. For a review of robust statistical methods in computer vision see [45, 46].

Note that there are two issues of robustness that must be addressed here. First, given the principal components, Black and Jepson [5] addressed the issue of robustly recovering the coefficients of a linear combination of basis vectors that reconstructs an input image (this step is what is commonly known as inference in the machine learning community). They did not address the general problem of robustly learning the principal components in the first place. Here we address the more general problem which involves learning both the basis vectors and linear coefficients robustly. Preliminary results of this work have been presented in [18].

2.1 Energy Functions and PCA

PCA is a statistical technique that is useful for dimensionality reduction. Let $\mathbf{D} = [\mathbf{d}_1\ \mathbf{d}_2\ ...\ \mathbf{d}_n] = [\mathbf{d}^1\ \mathbf{d}^2\ ...\ \mathbf{d}^d]^T$ be a matrix $\mathbf{D} \in \Re^{d \times n}$ [1], where each column $\mathbf{d}_i$ is a data sample (or image), n is the number of training images, and d is the number of pixels in each image. Previous formulations assume the data is zero mean. In the least-squares case, this can be achieved by subtracting the mean of the entire data set from each column $\mathbf{d}_i$. In the case of standard PCA we will consider the data be zero mean. For robust formulations, the "robust mean" must be explicitly estimated along with the principal components as described below.

Let the first k principal components of $\mathbf{D}$ be $\mathbf{B} = [\mathbf{b}_1, ..., \mathbf{b}_k] \in \Re^{d \times k}$. The columns of $\mathbf{B}$ are the directions of maximum variation within the data. The principal components maximize $\max_{\mathbf{B}} \sum_{i=1}^{n} ||\mathbf{B}^T\mathbf{d}_i||_2^2 = ||\mathbf{B}^T\mathbf{\Gamma}\mathbf{B}||_F$, with the constraint $\mathbf{B}^T\mathbf{B} = \mathbf{I}$, where $\mathbf{\Gamma} = \mathbf{D}\mathbf{D}^T = \sum_i \mathbf{d}_i\mathbf{d}_i^T$ is the covariance matrix. The columns of $\mathbf{B}$ form an orthonormal basis that spans the principal subspace. If the effective rank of $\mathbf{D}$ is much less than d and we can approximate the column space of $\mathbf{D}$ with $k << d$ principal components. The data $\mathbf{d}_i$ can be approximated as a linear combination of the principal components as $\mathbf{d}_i^{rec} = \mathbf{B}\mathbf{B}^T\mathbf{d}_i$ where $\mathbf{B}^T\mathbf{d}_i = \mathbf{c}_i$ are the linear coefficients obtained by projecting the training data onto the principal subspace; that is, $\mathbf{C} = [\mathbf{c}_1\ \mathbf{c}_2\ \dots\ \mathbf{c}_n] = \mathbf{B}^T\mathbf{D}$.

A method for calculating the principal components which is widely used in the statistics and neural network community [2, 20, 21, 51, 59] formulates PCA as the least-squares estimation of the basis images $\mathbf{B}$ that minimize:

$$\begin{aligned} E_{pca}(\mathbf{B}) &= \sum_{i=1}^{n} e_{pca}(\mathbf{e}_i) \\ &= \sum_{i=1}^{n} ||\mathbf{d}_i - \mathbf{B}\mathbf{B}^{\mathbf{T}}\mathbf{d}_i||_2^2 \\ &= \sum_{i=1}^{n}\sum_{p=1}^{d}(d_{pi} - \sum_{j=1}^{k} b_{pj}c_{ji})^2 \end{aligned} \quad (1)$$

[1] Bold capital letters denote a matrix $\mathbf{D}$, bold lower-case letters a column vector $\mathbf{d}$. $\mathbf{I}$ represents the identity matrix and $\mathbf{1}_m = [1, \cdots, 1]^T$ is a m-tuple of ones. $\mathbf{d}_j$ represents the j-th column of the matrix $\mathbf{D}$ and $\mathbf{d}^j$ is a column vector representing the j-th *row* of the matrix $\mathbf{D}$. d_{ij} denotes the scalar in row i and column j of the matrix $\mathbf{D}$ and the scalar i-th element of a column vector $\mathbf{d}_j$. d_{ji} is the i-th scalar element of the vector $\mathbf{d}^j$. All non-bold letters represent scalar variables. $diag$ is an operator that transforms a vector to a diagonal matrix, or a matrix into a column vector by taking each of its diagonal components. $[\mathbf{D}].^{-1}$ is an operator that calculates the inverse of each element of a matrix $\mathbf{D}$. $\mathbf{D}_1 \circ \mathbf{D}_2$ denotes the Hadamard (point wise) product between two matrices of equal dimension. $tr(\mathbf{A}) = \sum_{i=1}^{d} a_{ii}$ is the trace operator for a square matrix $\mathbf{A} \in \Re^{d \times d}$. $||\mathbf{D}||_F = tr(\mathbf{D}^T\mathbf{D}) = tr(\mathbf{D}\mathbf{D}^T)$ denotes the Frobenius norm of a matrix. Given a subspace F, $\dim(F)$, denotes the dimension of the subspace.

where $c_{ji} = \sum_{t=1}^{d} b_{tj} d_{ti}$, $\mathbf{B}^T\mathbf{B} = \mathbf{I}$ $||.||_2$ denotes the L_2 norm, $\mathbf{e}_i = \mathbf{d}_i - \mathbf{B}\mathbf{B}^T\mathbf{d}_i$ is the reconstruction error vector, and $e_{pca}(\mathbf{e}_i) = \mathbf{e}_i^T\mathbf{e}_i$ is the reconstruction error of $\mathbf{d}_i$.

Alternatively, we can make the linear coefficients explicit variables and minimize

$$E_{pca_2}(\mathbf{B}, \mathbf{C}) = \sum_{i=1}^{n} ||\mathbf{d}_i - \mathbf{B}\mathbf{c}_i||_2^2. \tag{2}$$

One approach for estimating both the bases, $\mathbf{B}$, and coefficients, $\mathbf{C}$, uses criss-cross regression [25], and it can be seen as a particular case of the Expectation Maximization (EM) algorithm used in Probabilistic PCA (PPCA) [57, 65]. PPCA assumes that the data is generated from a noisy random process and it defines a proper likelihood model. However, if the noise becomes infinitesimal and equal in all the directions PPCA becomes equivalent to standard PCA. In that case, the EM algorithm can be reduced to the following coupled equations [57]:

$$\mathbf{B}^T\mathbf{B}\mathbf{C} = \mathbf{B}^T\mathbf{D} \quad \text{(“E”-step)}, \tag{3}$$

$$\mathbf{B}\mathbf{C}\mathbf{C}^T = \mathbf{D}\mathbf{C}^T \quad \text{(“M”-step)}. \tag{4}$$

The algorithm alternates between solving for the linear coefficients $\mathbf{C}$ (“Expectation” step) and solving for the basis $\mathbf{B}$ (“Maximization” step). Although equations (3-4) have a “closed-form” solution in terms of an eigen-equation (the bases $\mathbf{B}$ are eigenvectors of the covariance matrix $\mathbf{D}\mathbf{D}^T$ [23, 38, 44]), for high dimensional data the EM approach can be more efficient in space and time [57, 65]. A more complex noise model is used in factor analysis, which assumes diagonal noise ($e_{pi} \sim N(0, \sigma_p^2),\ \forall p, i$) and that the coefficients $\mathbf{c}$ are Gaussian distributed with unit variance ($\mathbf{c} \sim N(\mathbf{0}, \mathbf{I})$) [44].

These Principal Component Analysis techniques have been extended to cope with the problem of missing data which occurs frequently in vision applications. Shum et al. [61] solve the PCA problem with *known* missing data by minimizing an energy function similar to (2) using a weighted least squares technique that ignores the missing data. The method is used to model a sequence of range images with occlusion and noise and is similar to the method of Gabriel and Zamir [25] described below. Also, Tenenbaum and Freeman [64] and Yuille et *al.* [72] use a similar trick to model missing data. Rao [55] proposed a Kalman filter approach for learning the bases $\mathbf{B}$ and the coefficients $\mathbf{C}$ in an incremental fashion. The observation process assumes Gaussian noise and corresponds the error E_{pca_2} above. While Rao does not use a robust learning method for

estimating the $\mathbf{B}$ and $\mathbf{C}$ that minimize E_{pca_2}, like Black and Jepson [5] he does suggest a robust rule for estimating the coefficients $\mathbf{C}$ once the bases $\mathbf{B}$ have been learned.

2.2 Previous Robust Approaches

The above methods for estimating the principal components are not robust to outliers that are common in training data and that can arbitrarily bias the solution [11, 38, 56, 58, 70] (e.g. Figure 1). This happens because the energy functions (or the covariance matrix) are derived from a least-squares (L_2 norm) framework. While the robustness of PCA methods in computer vision has received little attention, the problem has been studied in the statistics [11, 35, 38, 58] and neural networks [39, 70, 71] literature, and several algorithms have been proposed.

One approach replaces the standard estimation of the covariance matrix, $\mathbf{\Gamma}$, with a robust estimator of the covariance matrix, $\mathbf{\Gamma}^*$, [11, 35, 58]. This formulation weights the mean and the outer products which form the covariance matrix. Calculating the eigenvalues and eigenvectors of this robust covariance matrix gives eigenvalues that are robust to sample outliers. The mean and the robust covariance matrix can be calculated as:

$$\boldsymbol{\mu} = \frac{\sum_{i=1}^{n} w_1(M_i^2)\mathbf{d}_i}{\sum_{i=1}^{n} w_1(M_i^2)}, \tag{5}$$

$$\mathbf{\Gamma}^* = \frac{\sum_{i=1}^{n} w_2(M_i^2)(\mathbf{d}_i - \boldsymbol{\mu})(\mathbf{d}_i - \boldsymbol{\mu})^T}{\sum_{i=1}^{n} w_2(M_i^2) - 1}, \tag{6}$$

where $w_1(M_i^2)$ and $w_2(M_i^2)$ are scalar weights, which are a function of the Mahalanobis distance $M_i^2 = (\mathbf{d}_i - \boldsymbol{\mu})\mathbf{\Gamma}^{*^{-1}}(\mathbf{d_i} - \boldsymbol{\mu})$ and $\mathbf{\Gamma}^*$ is iteratively estimated. Numerous possible weight functions have been proposed (e.g. Huber's weighting coefficients [35] or $w_2(M_i^2) = (w_1(M_i^2))^2$ [11]). These approaches however, weight entire data samples rather than individual pixels and hence are not appropriate for many vision applications. Another related approach would be to robustly estimate each element of the covariance matrix. This is not guaranteed to result in a positive definite matrix [11]. These methods, based on robust estimation of the full covariance matrix, are computationally impractical for high dimensional data such as images (note that just computing the covariance matrix requires $O(nd^2)$ operations) and in some practical applications it is difficult to gather sufficient training data to guarantee that the covariance matrix is full rank.

Alternatively, Xu and Yuille [70] have proposed an algorithm that generalizes the energy function (1), by introducing additional binary variables that are zero when a data sample (image) is

considered an outlier. They minimize

$$\begin{aligned} E_{xu}(\mathbf{B}, \mathbf{V}) &= \sum_{i=1}^{n} \left[V_i \, ||\mathbf{d}_i - \mathbf{B}\mathbf{B}^{\mathbf{T}}\mathbf{d}_i||_2^2 + \eta(1 - V_i) \right] \\ &= \sum_{i=1}^{n} \left[V_i (\sum_{p=1}^{d} (d_{pi} - \sum_{j=1}^{k} b_{pj} c_{ij})^2) + \eta(1 - V_i) \right] \end{aligned} \tag{7}$$

where $c_{ij} = \sum_{t=1}^{d} b_{tj} d_{ti}$. Each V_i in $\mathbf{V} = [V_1, V_2, ..., V_n]$ is a binary random variable. If $V_i = 1$ the sample $\mathbf{d}_i$ is taken into consideration, otherwise it is equivalent to discarding $\mathbf{d}_i$ as an outlier. The second term in (7) is a penalty term, or prior, that discourages the trivial solution where all V_i are zero. Given $\mathbf{B}$, if the energy, $e_{pca}(\mathbf{e}_i) = ||\mathbf{d}_i - \mathbf{B}\mathbf{B}^{\mathbf{T}}\mathbf{d}_i||_2^2$ is smaller than a threshold η, then the algorithm prefers to set $V_i = 1$ considering the sample $\mathbf{d}_i$ as an inlier and 0 if it is greater than or equal to η.

Minimization of (7) involves a combination of discrete and continuous optimization problems and Xu and Yuille [70] derive a mean field approximation to the problem which, after marginalizing the binary variables, can be solved by minimizing:

$$E_{xu}(\mathbf{B}) = -\sum_{i=1}^{n} \frac{1}{T} f_{xu}(\mathbf{e}_i, T, \eta) \tag{8}$$

where $\mathbf{e}_i = \mathbf{d}_i - \mathbf{B}\mathbf{B}^T\mathbf{d}_i$ and where $f_{xu}(\mathbf{e}_i, T, \eta) = log(1 + e^{-T(e_{pca}(\mathbf{e}_i) - \eta)})$ is a function that is related to robust statistical estimators [6]. The T can be varied as an annealing parameter in an attempt to avoid local minima.

The above techniques are of limited application in computer vision problems as they reject entire images as outliers. In vision applications, outliers typically correspond to small groups of pixels and we seek a method that is robust to this type of outlier yet does not reject the "good" pixels in the data samples. Gabriel and Zamir [25] give a partial solution. They propose a weighted Singular Value Decomposition (SVD) technique that can be used to construct the principal subspace. In their approach, they minimize:

$$E_{gz}(\mathbf{B}, \mathbf{C}) = \sum_{i=1}^{n} \sum_{p=1}^{d} w_{pi} (d_{pi} - (\mathbf{b}^p)^T \mathbf{c}_i)^2 \tag{9}$$

where, recall, $\mathbf{b}^p$ is a column vector containing the elements of the p-th *row* of $\mathbf{B}$. This effectively puts a weight, w_{pi} on every pixel in the training data. In related work, Greenacre [30] gives a partial

solution to the problem of factorizing matrices with *known* weighting data by introducing Generalized Singular Value Decomposition (GSVD). This approach applies when the known weights in (9) are separable; that is, one weight for each row and one for each column: $w_{pi} = w_p w_i$. The basic idea is to first "whiten" the data using the weights, perform SVD, and then un-whiten the bases (for a similar idea, see [36]). The benefit of this approach is that it takes advantage of efficient implementations of the SVD algorithm. The disadvantages are that the weights must somehow already be known and that individual pixel outliers are not allowed.

In the general robust case, where the weights are unknown and there may be a different weight at every pixel in every training image, there is no such solution that leverages SVD, [25, 30] and one must solve the minimization problem with "criss-cross regressions" which involve iteratively computing dyadic (rank 1) fits using weighted least squares. The approach alternates between solving for $\mathbf{b}^p$ or $\mathbf{c}_i$ while the other is fixed; this is similar to the EM approach [57, 65] but without a probabilistic interpretation.

In this spirit, Gabriel and Odorof [24] note how the quadratic formulation in (1) is not robust to outliers and propose making the rank 1 fitting process in (9) robust. They propose a number of methods to make the criss-cross regressions robust but they apply the approach to very low-dimensional data and their optimization methods do not scale well to very high-dimensional data such as images. In related work, Croux and Filzmoser [16] use a similar idea to construct a robust matrix factorization based on a weighted L_1 norm. In the context of neural networks, Cichocki et *al.* [14] have proposed a sequential method for computing principal components robustifying Equation (1). Also, recently Skocaj et *al.* [63] proposed a Robust PCA algorithm similar in spirit to the work presented here, but they treat outliers as missing data and add a term to encourage their spatial coherence. In the following section, we develop these approaches further and give a complete solution that estimates all the parameters of interest, we connect the method with robust M-estimation techniques and unify previous results.

3 Robust Principal Component Analysis (RPCA)

In this section we extend previous work on robust PCA methods by adding an intra-sample outlier process motivated by the necessity of dealing with the type of outliers that typically occur in image

data.

The previous approach of Xu and Yuille (Equation (8)) suffers from three main problems: First, a single "bad" pixel value can make an image lie far enough from the subspace that the entire sample is treated as an outlier (i.e. $V_i = 0$) and has no influence on the estimate of $\mathbf{B}$. Second, Xu and Yuille use a least squares projection of the data $\mathbf{d}_i$ for computing the distance to the subspace; that is, the coefficients that reconstruct the data $\mathbf{d}_i$ are $\mathbf{c}_i = \mathbf{B}^T\mathbf{d}_i$. These reconstruction coefficients can be arbitrarily biased by an outlier. Finally, a binary outlier process is used which either completely rejects or includes a sample. Below we introduce a more general analogue outlier process that has computational advantages and provides a connection to robust M-estimation.

To address these issues we reformulate (7) as

$$E_{rpca}(\mathbf{B}, \mathbf{C}, \boldsymbol{\mu}, \mathbf{L}) = \sum_{i=1}^{n}\sum_{p=1}^{d}\left[L_{pi}(\frac{\tilde{e}_{pi}^2}{\sigma_p^2}) + P(L_{pi})\right] \tag{10}$$

where $0 \leq L_{pi} \leq 1$ is now an analog outlier process that depends on both images and pixel locations and $P(L_{pi})$ is a penalty function. The error $\tilde{e}_{pi} = d_{pi} - \mu_p - \sum_{j=1}^{k} b_{pj}c_{ji}$ and $\boldsymbol{\sigma} = [\sigma_1\ \sigma_2\ ...\ \sigma_d]^T$ specifies a "scale" parameter for each of the d pixel locations.

Observe that we explicitly solve for the mean $\boldsymbol{\mu}$ in the estimation process. In the least-squares formulation the mean can be computed in closed form and can be subtracted from each column of the data matrix $\mathbf{D}$. In the robust case, outliers are defined with respect to the error in the reconstructed images which include the mean. The mean can no longer be computed by performing a "deflation"[2] procedure, instead it is estimated (robustly) analogously to the other bases.

Also, recall that PCA assumes an isotropic noise model. In the formulation here we allow the noise to vary for every row (pixel) of the data ($e_{pi} \sim N(0, \sigma_p^2)$).

Exploiting the relationship between outlier processes and the robust statistics [6], minimizing (10) is equivalent to minimizing the following robust energy function:

$$\begin{aligned} E_{rpca}(\mathbf{B}, \mathbf{C}, \boldsymbol{\mu}, \boldsymbol{\sigma}) &= \sum_{i=1}^{n} e_{rpca}(\mathbf{d}_i - \boldsymbol{\mu} - \mathbf{B}\mathbf{c}_i, \boldsymbol{\sigma}) \\ &= \sum_{i=1}^{n}\sum_{p=1}^{d}\rho(d_{pi} - \mu_p - \sum_{j=1}^{k} b_{pj}c_{ji}, \sigma_p) \end{aligned} \tag{11}$$

[2]This technique is applied for efficiently computing eigenvectors by iteratively estimating one eigenvector and removing its influence from the data [20, 53] while making all the remaining eigenvectors orthogonal to it.

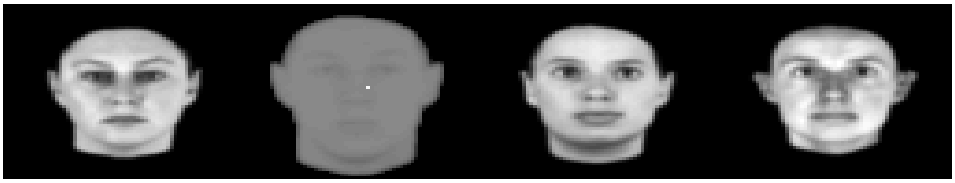


Figure 3: Original training Images. The second one is the log of original image.

for a particular class of robust ρ-functions [6]. We define the robust magnitude of a vector $\mathbf{x}$, as the sum of the robust error values for each component; that is, $e_{rpca}(\mathbf{x}, \boldsymbol{\sigma}) = \sum_{p=1}^{d} \rho(x_p, \sigma_p)$, where $\mathbf{x} = [x_1\ x_2\ ...\ x_d]^T$. Throughout the paper, we use the Geman-McClure error function [27] given by $\rho(x, \sigma_p) = \frac{x^2}{x^2+\sigma_p^2}$, where σ_p is a "scale" parameter that controls the convexity of the robust function and is used for deterministic annealing in the optimization process. This robust ρ-function corresponds to the penalty term $P(L_{pi}) = (\sqrt{L_{pi}} - 1)^2$ in (10) [6]. Many other choices can work well [6] but the Geman-McClure function has been used widely and has been shown to work well. Additionally, unlike some other ρ-functions, it is twice differentiable which is useful for optimization methods based on gradient descent. Details of the method are described below.

Note that while there are robust methods such as RANSAC and Least Median Squares ([45, 56]) that are theoretically more robust than M-estimation, it is not clear how to apply these methods efficiently to high dimensional problems such as the robust estimation of basis images.

3.1 Quantitative Comparison

In order to better understand how PCA and the method of Xu and Yuille are influenced by intra-sample outliers, we consider the contrived example in Figure (3), where four face images are shown. The second image is contaminated with one outlying pixel which has 10 times more energy than the sum of the others image pixels. In order to visualize the large range of pixel magnitudes, we have displayed the log of the image.

In Figure 4, the three learned bases given by standard PCA, Xu and Yuille's method, and our proposed method are shown. We force each method to explain the data using three basis images. Note that the approach of Xu and Yuille does not solve for the mean, hence, for a fair comparison we neither solved for nor subtracted the mean for any of the methods. In this case the mean is approximately recovered as one of the bases. The PCA bases capture the outlier in the second training image as the first principal component since it has the most energy. The other two bases

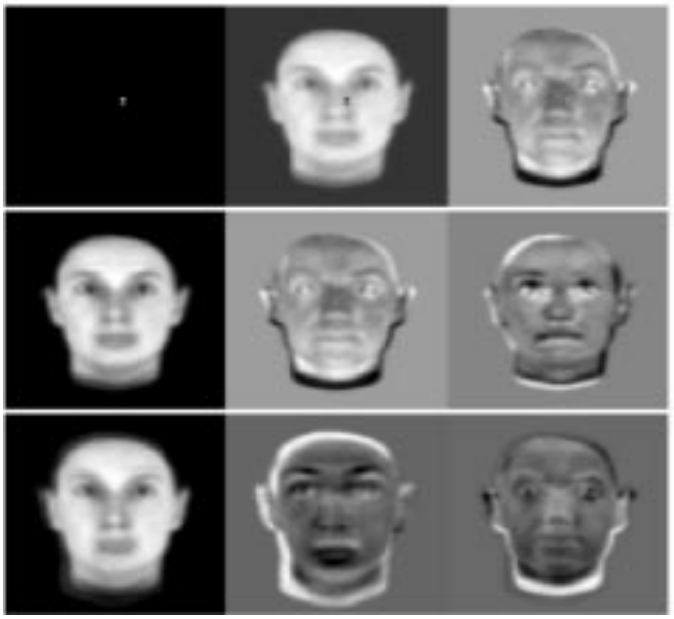


Figure 4: Learned basis images. *Top:* Traditional PCA. *Middle:* Xu and Yuille's method. *Bottom:* RPCA.

approximately capture the principal subspace spanning the other three images. Xu and Yuille's method, on the other hand, discards the second image for being far from the subspace and uses all three bases to represent the three remaining images. The RPCA method proposed here, constructs a subspace that takes into account all four images while ignoring the single outlying pixel. Hence, we recover three bases to approximate the four images.

In Figure 5 we project the original images (without outliers) onto the three learned basis sets. PCA "wastes" one of its three basis images on the outlying data and hence has only two basis images to approximate four training images. Xu and Yuille's method ignores all the useful information in image 2 as the result of a single outlier and, hence, is unable to reconstruct that image. Since it uses three basis images to represent the other three images, it can represent them perfectly. The RPCA method provides an approximation of all four images with three basis images. The MSRE (MSRE=$\frac{1}{n}\sum_{i=1}^{n}||\mathbf{d}_i - \boldsymbol{\mu} - \mathbf{B}\mathbf{c}_i||_2^2$) is less for RPCA than for the other methods; the RPCA error is 7.02, while PCA and Xu and Yuille's method give errors of 18.59 and 9.02 respectively.

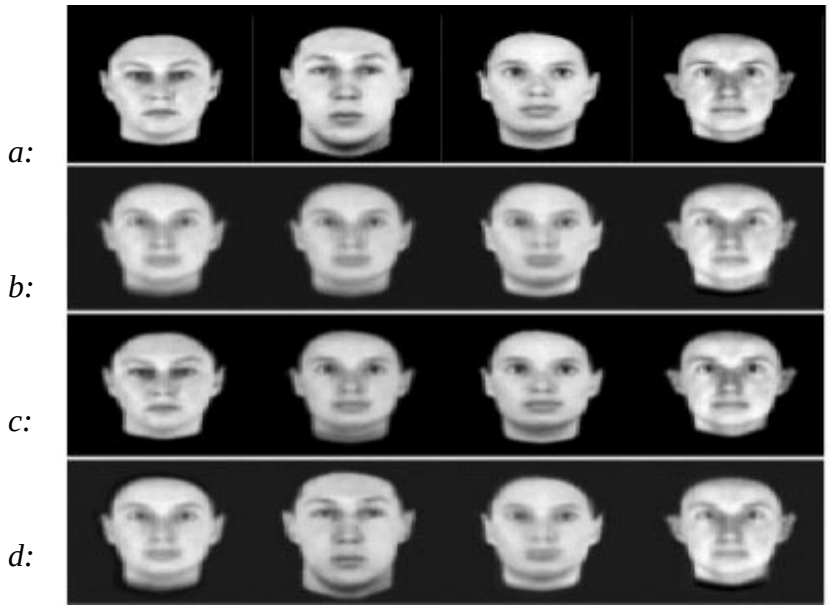


Figure 5: Reconstruction from noiseless images. *(a).* Original images. *(b).* Noiseless reconstruction using PCA bases. *(c).* Using Xu and Yuille's method. *(d).* Using RPCA.

3.2 Computational Issues

In this section, we describe how to robustly compute the mean and the subspace spanned by the first k principal components. We do this without imposing orthogonality between the bases; this can be imposed later if needed [65] or with a Gramm-Schmidt procedure [59]. The section is organized as follows: Section 3.2.1 will introduce an iteratively re-weighted least-squares approach to solve Equation (11), this will allow us to relate this method to previous work and will provide insight into the problem. Section 3.2.2 examines some special cases of the problem that admit closed form solutions. Section 3.2.3 derives a continuous optimization formulation that results in a more efficient algorithm and is particularly useful for high dimensional image data. Section 3.2.4 shows how to compute the scale parameter, σ, in the robust function automatically. Finally section 3.2.5 discusses several practical issues such as initialization, selection of the number of bases, and criteria for convergence.

3.2.1 A Weighted Least-Squares problem

As we have seen in the previous section, robust problems, in general, can be posed as the minimization of an energy function or cost function. While many optimization methods exist, it is instructive

and useful to formulate the minimization of Equation (11) as a weighted least squares problem and solve it using iteratively reweighted least-squares (IRLS) [3, 34, 42]. In particular, this will provide insight into the relationships between RPCA and previous methods. Originally proposed by Beaton and Turkey [3] and used widely in statistics [34, 42] and computer vision [41, 73], IRLS [3, 34, 42] is an approximate and iterative algorithm for solving robust M-estimation problems.

We define the residual error in matrix notation as, $\tilde{\mathbf{E}} = \mathbf{D} - \boldsymbol{\mu}\mathbf{1}_n^T - \mathbf{BC}$. Then, for a given $\boldsymbol{\sigma}$, a matrix $\mathbf{W} \in \Re^{d \times n}$ can be defined such that it contains positive weights for each pixel and each image. $\mathbf{W}$ is calculated for each iteration as a function of the previous residuals $\tilde{e}_{pi} = d_{pi} - \mu_p - \sum_{j=1}^{k} b_{pj} c_{ji}$ and it is related to the "influence" [32] of pixels on the solution. Each element, w_{pi}, of $\mathbf{W}$ will be equal to $w_{pi} = \psi(\tilde{e}_{pi}, \sigma_p)/\tilde{e}_{pi}$, where $\psi(\tilde{e}_{pi}, \sigma_p) = \frac{\partial \rho(\tilde{e}_{pi}, \sigma_p)}{\partial \tilde{e}_{pi}} = \frac{2\tilde{e}_{pi}\sigma_p^2}{(\tilde{e}_{pi}^2 + \sigma_p^2)^2}$ for the Geman-McClure ρ-function. As in previous work on W-estimators [3, 34, 42], robust M-estimation with a ρ-function like the one here, can be solved using IRLS. For an iteration of IRLS, Equation (11), can be transformed into a weighted least-squares problem and rewritten as:

$$E(\mathbf{B}, \mathbf{C}, \boldsymbol{\mu}, \mathbf{W})_{wpca} = \sum_{i=1}^{n} (\mathbf{d}_i - \boldsymbol{\mu} - \mathbf{Bc}_i)^T \mathbf{W}_i (\mathbf{d}_i - \boldsymbol{\mu} - \mathbf{Bc}_i) \quad (12)$$

$$= \sum_{p=1}^{d} (\mathbf{d}^p - \mu_d \mathbf{1}_n - \mathbf{C}^T \mathbf{b}^p)^T \mathbf{W}^p (\mathbf{d}^p - \mu_d \mathbf{1}_n - \mathbf{C}^T \mathbf{b}^p) \quad (13)$$

where the $\mathbf{W}_i \in \Re^{d \times d} = diag(\mathbf{w}_i)$ are diagonal matrices containing the positive weighting coefficients for the data sample $\mathbf{d}_i$, and recall that $\mathbf{w}_i$ is the i^{th} column of $\mathbf{W}$. $\mathbf{W}^p \in \Re^{n \times n} = diag(\mathbf{w}^p)$ are diagonal matrices containing the weighting factors for the p^{th} pixel over the whole training set. Note the symmetry of (12) and (13) where, recall, $\mathbf{d}_i$ represents the i^{th} column of the data matrix $\mathbf{D}$ and $\mathbf{d}^p$ is a column vector which contains the p^{th} row. Observe that (12) and (13) have non-unique solutions since, for any linear invertible transformation matrix $\mathbf{R}$, $\mathbf{BRR}^{-1}\mathbf{C}$ would give the same solution (i.e. the reconstruction from the subspace will be the same). This ambiguity can be solved by imposing the constraint of orthogonality between the bases $\mathbf{B}^T\mathbf{B} = \mathbf{I}$ (e.g. with Graham-Schmidt orthogonalization). There are, however, many computer vision applications (e.g subspace methods for recognition) in which the important measurement is the distance from the subspace and the particular axes of the subspace are irrelevant.

In order to find a solution to $E(\mathbf{B}, \mathbf{C}, \boldsymbol{\mu}, \mathbf{W})_{wpca}$, we differentiate (12) w.r.t. $\mathbf{c}_i$ and $\boldsymbol{\mu}$ and differentiate (13) w.r.t. $\mathbf{b}^p$ to find necessary, but not sufficient conditions for the minimum. From

these conditions, we derive the following coupled system of equations,

$$\boldsymbol{\mu} = (\sum_{i=1}^{n} \mathbf{W}_i{}^{-1}) \sum_{i=1}^{n} \mathbf{W}_i(\mathbf{d}_i - \mathbf{B}\mathbf{c}_i), \tag{14}$$

$$(\mathbf{B}^T\mathbf{W}_i\mathbf{B})\mathbf{c}_i = \mathbf{B}^T\mathbf{W}_i(\mathbf{d}_i - \boldsymbol{\mu}) \ \forall\, i = 1 \ldots n, \tag{15}$$

$$(\mathbf{C}\mathbf{W}^j\mathbf{C}^T)\mathbf{b}^j = \mathbf{C}\mathbf{W}^j(\mathbf{d}^j - \mu_d\mathbf{1}_n) \ \forall\, j = 1 \ldots d. \tag{16}$$

Giving these updates of the parameters, an approximate algorithm for minimizing Equation (11) can employ a two step method that minimizes $E_{wpca}(\mathbf{B}, \mathbf{C}, \boldsymbol{\mu})$ using Alternated Least Squares (ALS) (or criss-cross regressions [25]).

Summarizing, the whole IRLS procedure works as follows, first an initial basis $\mathbf{B}^{(0)}$ and a set of coefficients $\mathbf{C}^{(0)}$ are given[3], then the initial error, $\tilde{\mathbf{E}}^{(0)}$, can be calculated along with the $\boldsymbol{\sigma}$ parameters (as described below). Then the weighting matrix $\mathbf{W}^{(1)}$ can be computed and it will be used to successively alternate between minimizing with respect to $\mathbf{c}_i^{(1)}$ and $(\mathbf{b}^j)^{(1)}$ $\forall i, j$ and $\boldsymbol{\mu}^{(1)}$ in closed form using Equations (14), (15), and (16). Once $\mathbf{c}_i^{(1)}, (\mathbf{b}^j)^{(1)}, \boldsymbol{\mu}^{(1)}$ have converged, we recomputed the error, $\tilde{\mathbf{E}}^{(1)}$ and calculate the weighting matrix, $\mathbf{W}^{(2)}$, then we proceed in the same manner until convergence of the algorithm. Also, during this process we anneal $\boldsymbol{\sigma}$. At this point it is worth noting that there are several possible ways to update the parameters more efficiently, rather than a closed form solution, see for instance [3, 34, 42].

3.2.2 Two special cases

In the general case (with arbitrary weighting matrices) is not clear that there exists a closed form solution in terms of a weighted covariance matrix. For instance, consider the simple scenario in which the mean is zero, $\boldsymbol{\mu} = \mathbf{0}$, the weights $\mathbf{W}$ are known, and our goal is to compute just the first weighted eigenvector and coefficient; that is $\mathbf{B} = [\mathbf{b}^1]^T = \mathbf{b}_1$ and $\mathbf{C} = [\mathbf{c}^1]^T = [c_1, c_2, \ldots, c_n]$ Observe that, in this simple scenario, the energy function (12) can be expressed as the following quotient, $E(\mathbf{b}_1)_{wpca} = \sum_i \frac{(\mathbf{b}_1^T\mathbf{W}_i\mathbf{d}_i)^2}{\mathbf{b}_1^T\mathbf{W}_i\mathbf{b}_1}$. But due the fact that the normalization factor depends on $\mathbf{W}_i$, we cannot solve this equation in terms of an eigen-equation. However, it is worth mentioning two interesting special cases that have solutions in terms of the eigenvectors of a weighted covariance matrix and that can be very useful for practical applications. These problems can be solved using Generalized Singular Value Decomposition (GSVD) [31].

[3]The number between parenthesis indicates the iteration number.

First, if every sample (image) has a different weight, w_i, then $\mathbf{W}_i = w_i\mathbf{I}$. By imposing the constraint $\mathbf{B}^T\mathbf{B} = \mathbf{I}$, it is easy to show that $E(\mathbf{B})_{wpca} = \mathbf{B}^T(\sum_i w_i\mathbf{d}_i\mathbf{d}_i^T)\mathbf{B}$. This is useful when each image may have its own weight; for example, if data is collected sequentially, then one might want to weight more recent data higher than old data. The solution to this problem is given by the eigenvectors of the weighted covariance matrix $\sum_i w_i\mathbf{d}_i\mathbf{d}_i^T$. Note that this particular case, with just one weight for each sample, is analogous to previous robust PCA methods based on robust covariance matrices [11, 58] and to the method of Xu and Yuille [69]. This GSVD formulation provides a simple (and efficient) algorithm for this class of problems *provided that the weights are known.* Also observe, that in this case, Equation (15) becomes a regular least-squares projection, since the entire sample has the same weight, and just the rows of the bases $(\mathbf{b}^j)$ are weighted in (16)[4].

The second special case occurs when some fixed weight matrix, $\hat{\mathbf{W}}$, applies to all the images in the training set; that is, $\mathbf{W}_i = \hat{\mathbf{W}}$. This occurs, for example with systematic missing data where we would like to weight pixel locations with a binary value indicating whether or not the training set contains data at that pixel. It may also be useful to have a particular spatial weight matrix for a given application such as face modeling where, for example, we might give more weight to the eyes and mouth (see Figure 7) to obtain a more accurate reconstruction in those areas. In this case the solution can be found by minimizing the Rayleigh quotient $E(\mathbf{B})_{wpca} = \sum_i \frac{||\mathbf{B}^T\mathbf{\Gamma}\mathbf{B}||_2^2}{||\mathbf{B}^T\hat{\mathbf{W}}\mathbf{B}||_2^2}$, were $\mathbf{\Gamma} = \hat{\mathbf{W}}(\sum_{i=1}^n \mathbf{d}_i\mathbf{d}_i^T)\hat{\mathbf{W}}$. The solution is the well known generalized eigenvalue problem, $\mathbf{\Gamma B} = \hat{\mathbf{W}}\mathbf{B\Lambda}$. It is interesting to observe, as a dual property of the previous case, that (16) becomes a least-squares estimation problem; that is, in the computation of the bases, no weights are involved and just the coefficients $\mathbf{c}_i$ are calculated with weighted information.

3.2.3 Updating parameters

For solving the more general case, where the weights can be different for every pixel in every image, we can employ a two step method that minimizes $E_{wpca}(\mathbf{B}, \mathbf{C}, \boldsymbol{\mu})$ as explained in section 3.2.1. The most computationally expensive part of the algorithm involves computing (15) and (16). The computational cost of one iteration of (15) is $O(nk^2d) + O(nk^3) + O(nkd)$ for $\mathbf{C}$ and

[4] In this case, an efficient algorithm can exploit the fact that all the matrices in the linear system of equations are the same and for all the rows we can simultaneously solve all the systems of equations.

$O(nk^2d) + O(dk^3) + O(nkd)$ for $\mathbf{B}$ (16). Typically $d \gg n \gg k$, and, for example, estimating the bases $\mathbf{B}$ involves computing the solution of d systems of $k \times k$ equations, which for large d is computationally expensive. Rather than directly solving d systems of $k \times k$ (16) or n systems of $k \times k$ (15), we solve for $\mathbf{B}$, $\mathbf{C}$ and $\boldsymbol{\mu}$ using gradient descent with a local quadratic approximation to determine an estimation of the step sizes (see [5, 10] for further information). The robust learning "rules" for updating successively $\mathbf{B}$, $\mathbf{C}$ and $\boldsymbol{\mu}$ are then as follows:

$$\mathbf{B}^{(n+1)} = \mathbf{B}^{(n)} - [\mathbf{H_b}].^{-1} \circ \frac{\partial E_{rpca}}{\partial \mathbf{B}} \tag{17}$$

$$\mathbf{C}^{(n+1)} = \mathbf{C}^{(n)} - [\mathbf{H_c}].^{-1} \circ \frac{\partial E_{rpca}}{\partial \mathbf{C}} \tag{18}$$

$$\boldsymbol{\mu}^{(n+1)} = \boldsymbol{\mu}^{(n)} - [\mathbf{H}_{\boldsymbol{\mu}}].^{-1} \circ \frac{\partial E_{rpca}}{\partial \boldsymbol{\mu}} \tag{19}$$

where, recall, $[\mathbf{H}].^{-1}$ is an element-wise inverse of a matrix $\mathbf{H}$. The partial derivatives with respect to the parameters are:

$$\frac{\partial E_{rpca}}{\partial \mathbf{B}} = -\boldsymbol{\Psi}(\tilde{\mathbf{E}}, \boldsymbol{\sigma})\mathbf{C}^T \tag{20}$$

$$\frac{\partial E_{rpca}}{\partial \mathbf{C}} = -\mathbf{B}^T\boldsymbol{\Psi}(\tilde{\mathbf{E}}, \boldsymbol{\sigma}) \tag{21}$$

$$\frac{\partial E_{rpca}}{\partial \boldsymbol{\mu}} = -\boldsymbol{\Psi}(\tilde{\mathbf{E}}, \boldsymbol{\sigma})\mathbf{1}_n \tag{22}$$

where $\tilde{\mathbf{E}}$ is the reconstruction error and an estimate of the step size is given by:

$$\mathbf{H_b} = \boldsymbol{\zeta}(\tilde{\mathbf{E}}, \boldsymbol{\sigma})(\mathbf{C} \circ \mathbf{C})^T \qquad \mathbf{h_b}_i = max\ \ diag\Big(\frac{\partial^2 E_{rpca}}{\partial \mathbf{b}_i \partial \mathbf{b}_i^T}\Big) \tag{23}$$

$$\mathbf{H_c} = (\mathbf{B} \circ \mathbf{B})^T\boldsymbol{\zeta}(\tilde{\mathbf{E}}, \boldsymbol{\sigma}) \qquad \mathbf{h_c}_i = max\ \ diag\Big(\frac{\partial^2 E_{rpca}}{\partial \mathbf{c}_i \partial \mathbf{c}_i^T}\Big) \tag{24}$$

$$\mathbf{H}_{\boldsymbol{\mu}} = \boldsymbol{\zeta}(\tilde{\mathbf{E}}, \boldsymbol{\sigma})\mathbf{1}_n \qquad \mathbf{h}_{\boldsymbol{\mu}_i} = max\ \ diag\Big(\frac{\partial^2 E_{rpca}}{\partial \boldsymbol{\mu} \partial \boldsymbol{\mu}^T}\Big) \tag{25}$$

where $\frac{\partial E_{rpca}}{\partial \mathbf{B}} \in \Re^{d \times k}$ is the derivative of E_{rpca} with respect to $\mathbf{B}$, and similarly for $\frac{\partial E_{rpca}}{\partial \mathbf{C}} \in \Re^{k \times n}$ and $\frac{\partial E_{rpca}}{\partial \boldsymbol{\mu}} \in \Re^{d \times 1}$. $\boldsymbol{\Psi}(\tilde{\mathbf{E}}, \boldsymbol{\sigma}) = \mathbf{W} \circ \tilde{\mathbf{E}} \in R^{d \times n}$ is a matrix that contains the derivatives of the robust ρ-function (i.e. each element pi is $\psi(\tilde{e}_{pi}, \sigma_p) = w_{pi}\tilde{e}_{pi}$). $\mathbf{H_b} \in \Re^{d \times k}$ is a matrix in which every component h_{ij_b} is an upper bound of the second derivative w.r.t. $\mathbf{b}$; that is, $h_{ij_b} \geq \frac{\partial^2 E_{rpca}}{\partial b_{ij}^2}$. Each element ζ_{pi} of the matrix $\boldsymbol{\zeta}(\tilde{\mathbf{E}}, \boldsymbol{\sigma}) \in R^{d \times n}$, contains a maximum of the second derivative [10]; that is, $\zeta_{pi} = \max_{\tilde{e}_{pi}} \frac{\partial^2 \rho(\tilde{e}_{pi}, \sigma_p)}{\partial \tilde{e}_{pi}^2} = \frac{2}{\sigma_p^2}$. Recall that $\mathbf{h_b}_i$ is the i^{th} column of the matrix $\mathbf{H_b}$,

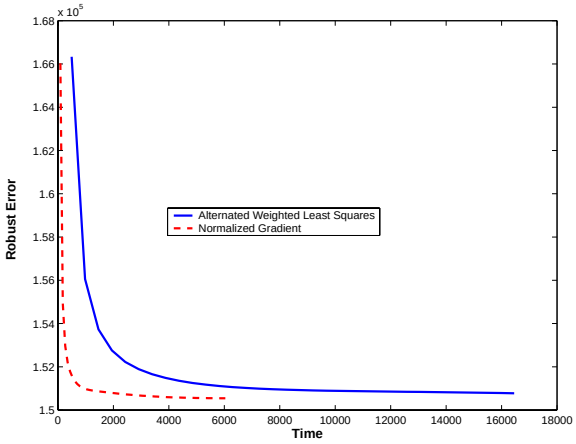


Figure 6: Robust Error versus computation time (seconds). The experiment uses 300 images of 120×160 pixels and involves computing 30 bases. The graph plots reconstruction error versus computation time for both gradient descent and alternated least squares approaches.

which is computed by taking a maximum of the diagonal of the Hessian matrix for this column. The same can be applied to the matrices $\mathbf{H}_{\mathbf{c}}$ and $\mathbf{H}_{\boldsymbol{\mu}}$. After each update of $\mathbf{B}$, $\mathbf{C}$, or $\boldsymbol{\mu}$, we update the error $\tilde{\mathbf{E}}$. Also several iterations for each update of $\mathbf{B}$, $\mathbf{C}$ and $\boldsymbol{\mu}$ are possible.

Now, the computational cost of one iteration of the updating of $\mathbf{B}$, $\mathbf{C}$ or $\boldsymbol{\mu}$ with the normalized gradient descent is linear in all the parameters, that is, $O(ndk)$. For testing these approaches, we implemented in Matlab both updating schemes and compare their convergence properties. Figure (6) plots the energy (E_{rpca}) versus the computation time for the normalized gradient descent and compares this with ALS. As can be observed the gradient descent algorithm approaches the local minimum faster, although it typically requires more (but less computationally expensive) iterations to converge. Another benefit of the normalized gradient updating rule is that it allows incremental or on-line learning [70]. Performing on-line learning is of particular interest when data becomes available over time and one must re-estimate the parameters to account for the new data. Also this incremental update algorithm can be useful when, due to memory limitations, not all the data can be loaded into memory. In such a case one can iteratively load different subsets of the original data and adapt the model parameters; the convergence properties of this method are not addressed here.

3.2.4 Local measure of the scale value

The scale parameter $\boldsymbol{\sigma}$ controls the shape of the robust ρ-function and hence determines what residual errors are treated as outliers. When the absolute value of the robust error $|\tilde{e}_{pi}|$ is larger

Figure 7: Local σ_p values estimated in 4×4 regions for a training set of 100 face images.

than $\frac{\sigma_p}{\sqrt{3}}$, the ρ-function used here begins reducing the influence of the pixel p in image i on the solution [6]. We estimate the scale parameters σ_p, for each pixel p, automatically using the Median Absolute Deviation (MAD) [7, 34, 42, 56] of the pixel (although other approaches are possible). The MAD can be viewed as a robust statistical estimate of the standard deviation, and we compute it as:

$$\sigma_p = \beta \max(1.4826 \, \text{med}_R(|\mathbf{e}^p - \text{med}_R(|\mathbf{e}^p|)|), \sigma_{min}) \tag{26}$$

where med_R indicates that the median is taken over a region, R, around pixel p and σ_{min} is the MAD over the whole image [7]. β is a constant factor that sets the outlier σ_p to be between 1 and 2.5 times the estimated deviation.

For calculating the MAD, we need to have an initial error, $\mathbf{e}^p$, which is obtained as follows: we compute the standard PCA of the data, and calculate the number of bases which preserve 55% of the energy (E_{pca}). This is achieved when the ratio between the energy of the reconstructed vectors and the original ones is larger than 0.55; that is, $\xi = \frac{\sum_{i=1}^{n} ||\mathbf{B}\mathbf{c}_i||_2^2}{\sum_{i=1}^{n} ||\mathbf{d}_i||_2^2} \geq 0.55$. Observe, that with standard PCA, this ratio can be calculated in terms of eigenvalues of the covariance matrix [20]. With this number of bases we compute the least-squares reconstruction error $\mathbf{E}$ and use that to obtain a robust estimate of $\boldsymbol{\sigma}$. Although other methods have been proposed in the statistical literature [32, 35] that recalculate the $\boldsymbol{\sigma}$ at each iteration, we found this method very stable.

Figure 7 shows the local σ_p values for the training set in Figure 1. Observe how larger values of σ_p are estimated for the eyes, mouth, and boundary of the face. This indicates that there is higher variance in the training set in these regions and larger deviations from the estimated subspace should be required before a training pixel is considered an outlier.

3.2.5 Initialization and other issues

Since minimization of (11) is an iterative scheme, an initial guess for the parameters $\mathbf{B}$ or $\mathbf{C}$ and $\boldsymbol{\mu}$ has to be given. An initial estimate of $\boldsymbol{\mu}$ is given by the robust mean which can be found by minimizing $E_{rmean}(\overline{\mathbf{x}}) = \sum_i e_{rpca}(\mathbf{d}_i - \overline{\mathbf{x}}, \boldsymbol{\sigma})$. Alternatively, simply taking the median or mean is often sufficient. As an initial guess for $\mathbf{B}$ we chose the standard principal components. The parameters $\mathbf{C}$ are just the solution of the linear system of equations which minimize $min_{\mathbf{C}}||\mathbf{D} - \boldsymbol{\mu}\mathbf{1}_n^T - \mathbf{BC}||_2$. With these parameters and $\boldsymbol{\sigma}_{initial}$ we calculate the $\mathbf{W}$ which starts the process.

In general the energy function (11) is non-convex and the minimization method can get trapped in local minima. We make use of a deterministic annealing scheme which helps avoid these local minima [5]. The method begins with $\boldsymbol{\sigma}$ being a large multiple of (26) such that all pixels are inliers. Then $\boldsymbol{\sigma}$ is successively lowered to the value given by (26), reducing the influence of outliers. While it is not guaranteed to converge to a global minimum, experimental results have shown reasonable convergence points. If one is concerned about local minima, the algorithm can be run multiple times with different initial conditions. The solution with the lowest minimum error can then be chosen. In practice, with reasonable initial estimates, the algorithm converges to similar results (visually and in terms of robust error).

Since the method is iterative in nature, it is necessary to impose some termination criterion. Several methods can be chosen (e.g. that the difference between two or more successive errors is less than a threshold, the norm of two or more consecutive updates of the parameters is less than a certain parameter ϵ, etc). However, since the method should converge to a subspace, we found that a good stopping criterion can be defined in terms of the principal angles [28] between two consecutive subspaces $\mathbf{B}^{(n)}$ and $\mathbf{B}^{(n+1)}$. The largest principal angle is related to the "distance" between equidimensional subspaces. These principal angles can be computed efficiently with the QR factorization and the SVD algorithm [28]. If the principal angle is smaller than a certain ϵ or we have reached a maximum number of iterations, the iterative procedure will stop (Figure (13*b*)).

With standard PCA, the number of bases is usually selected to preserve some percentage of the energy (E_{pca}); for example, $\frac{||\mathbf{BC}||_2^2}{||\mathbf{D}||_2^2} \geq 0.9$. With RPCA this criterion is not straightforward to apply particularly in the case of real problems with high dimensional data. The robust error, E_{rpca}, (11), depends on the $\boldsymbol{\sigma}$ and the number of bases so we can not directly compare energy functions with

different scale parameters. Moreover, the energy of the outliers is *confused* with the energy of the signal. We have experimented with different methods for automatically selecting the number of basis images including the Minimum Descriptor Length (MDL) criterion and Akaike Information Criterion (AIC) . However, these model selection methods do not scale well to high dimensional data and require the manual selection of a number of normalization factors. We have exploited more heuristic methods here that work well in practice.

We apply standard PCA to the data, and calculate the number of bases that preserve 55% of the energy (E_{pca}). With this number of bases, we apply RPCA, minimizing (11), until convergence. At the end of this process we have a matrix $\mathbf{W}$ that contains the weighting of each pixel in the training data. We detect outliers using this matrix and set the values of $\mathbf{W}$ to 0 if $|w_{pi}| > \frac{\sigma_p}{\sqrt{3}}$ and to w_{pi} otherwise, obtaining $\mathbf{W}^*$. The value $\frac{\sigma_p}{\sqrt{3}}$ represents the point at which the robust ρ-function begins down-weighting the contribution of data and, hence, can be thought of as an outlier rejection point [6]. We then incrementally add additional bases and minimize $E(\mathbf{B}, \mathbf{C}, \boldsymbol{\mu}) = ||\mathbf{W}^* \circ (\mathbf{D} - \boldsymbol{\mu}\mathbf{1}_n^T - \mathbf{BC})||_2^2$ with the same method as before but maintaining constant weights $\mathbf{W}^*$. We proceed adding bases until the percentage of energy accounted for, ξ, is bigger than 0.9, where:

$$\xi = \frac{\sum_{i=1}^{n} (\mathbf{Bc}_i)^T \mathbf{W}_i^* (\mathbf{Bc}_i)}{\sum_{i=1}^{n} (\mathbf{d}_i - \boldsymbol{\mu})^T \mathbf{W}_i^* (\mathbf{d}_i - \boldsymbol{\mu})} \tag{27}$$

Once the linear subspace has been learned, images can be robustly reconstructed for a variety of applications [5]. In order to robustly compute the coefficients $\mathbf{c}_i$ of a new given image $\mathbf{d}_i$, we first subtract the robust mean $\boldsymbol{\mu}$ and compute the coefficients $\mathbf{C}$ using Equation (15). Note that there is no need to also update the bases in this case. Note also that $\boldsymbol{\sigma}$ values used in the robust reconstruction are those learned during the training process.

The following pseudocode describes the whole optimization process:

- Compute the robust mean, $\boldsymbol{\mu}^{(0)}$, and the standard PCA solution (e.g. with the SVD). Calculate the residuals, initialize $\mathbf{B}^{(0)}, \mathbf{C}^{(0)}$ and select the initial number of bases.

- Calculate the scale parameter $\boldsymbol{\sigma}$ (Equation 26 (MAD)), this will be $\boldsymbol{\sigma}_{final}$. Multiply it by a constant, so all the pixels are inliers at the beginning, that is $\boldsymbol{\sigma}_{final} = K * \boldsymbol{\sigma}_{initial}$.

- Until the principal angle between $\mathbf{B}^{(n)}$ and $\mathbf{B}^{(n+1)}$ is less than a chosen ϵ:

- Compute an estimation of the step size (Equations (23), (24), (25)).
- Compute the partial derivatives w.r.t. the parameters (Equation 22).
- Update the parameters (Equations (17), (18), (19)).
- Lower $\boldsymbol{\sigma}$ according to the annealing schedule if it is bigger than $\boldsymbol{\sigma}_{final}$.

- Additionally, compute $\mathbf{W}^*$ by thresholding the weight matrix $\mathbf{W}$. Keep adding bases and solving for $\mathbf{B}$, $\mathbf{C}$, and $\boldsymbol{\mu}$ while Equation (27) is less than 0.9.

4 A note on Robust SVD

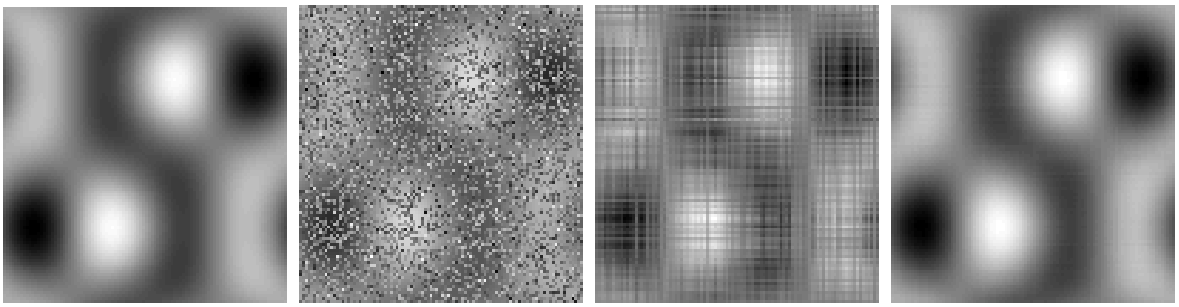
Singular Value Decomposition returns the factorization of a real matrix $\mathbf{D} \in \Re^{d \times n}$ into three matrices such that $\mathbf{D} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T$ where $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_s) \in \Re^{d \times n}$, $s = min\{d, n\}$, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_s \geq 0$. The matrices $\mathbf{U} \in \Re^{d \times d}$ and $\mathbf{V} \in \Re^{n \times n}$ are orthogonal and span the column and row space of $\mathbf{D}$ respectively.

SVD gives the best rank k approximation of a real matrix $\mathbf{D}$, that minimizes $||\mathbf{D} - \mathbf{U}\boldsymbol{\Lambda}\mathbf{V}^T||$ for *any* unitarily invariant norm (L_2 norm, Frobenius Norm, etc.) [47]. Observe at this point, that if we rename $\mathbf{C} = \boldsymbol{\Lambda}\mathbf{V}^T$, and we assume zero mean data in the PCA model, the subspace spanned by the matrix $\mathbf{B}$ of (2) and the matrix $\mathbf{U}$ of the SVD are the same (for the same number of bases). PCA and SVD both can be formulated as bilinear regression problems. Therefore, for performing Robust Singular Value Decomposition (RSVD), we will proceed in the same manner as RPCA. That is, given the number of bases k and the $\boldsymbol{\sigma}$ parameters, we calculate the robust subspace, $\mathbf{B} \in \Re^{d \times k}$, spanned by the first k bases, by minimizing $E(\mathbf{B}, \mathbf{C}) = \sum_{i=1}^{n} e_{rpca}(\mathbf{d}_i - \mathbf{B}\mathbf{c}_i, \boldsymbol{\sigma})$ as before.

Then, given $\mathbf{B}$ and $\mathbf{C}$, we robustly reconstruct the data $\mathbf{D} \approx \tilde{\mathbf{D}} = \mathbf{B}\mathbf{C}$ which effectively filters out the outlying data. Now given the reconstructed data, $\tilde{\mathbf{D}}$ that is free of outliers, we perform standard SVD to compute $\tilde{\mathbf{D}} = \mathbf{B}\mathbf{C} = \tilde{\mathbf{U}}\tilde{\boldsymbol{\Lambda}}\tilde{\mathbf{V}}^T$. Note that at most the first k singular values of $\tilde{\boldsymbol{\Lambda}}$ will be non-zero since the robust reconstructed subspace has dimension k; i.e. $dim(\mathbf{B}\mathbf{C}) = k$.

We take this approach of using RPCA since the method has already been developed above and is straightforward to apply here. An alternative approach consists of explicitly calculating $\mathbf{U}$, $\boldsymbol{\Lambda}$

Figure 8: Two sinusoidal plaid patterns and their linear combination.



a *b* *c* *d*

Figure 9: SVD Experiment. *a.* Original data. Each column of the matrix is a "sample" (i.e. a 1D image). *b.* Training data with the addition of outliers. *c.* Least squares (SVD) reconstruction of the training data. *d.* Robust SVD automatically removes the outliers and results in a reconstruction similar to the original data.

and $\mathbf{V}^T$ while imposing orthogonality on $\mathbf{U}$ and $\mathbf{V}^T$ (e.g. with Gramm-Schmidt orthogonalization).

To show the benefits of the RSVD, we synthetically generate a sinusoidal plaid pattern (Figure 8) of 100×100 pixels. The sinusoidal pattern is composed of the sum of the outer products of two unidimensional sinusoidal signals; that is, $\mathbf{D} = \sum_{i=1}^{2} \mathbf{b}_i (\mathbf{c}^i)^T = \mathbf{BC}$, where $\mathbf{B} \in \Re^{100 \times 2}$ and $\mathbf{C} \in \Re^{2 \times 100}$. In the two matrices on the left hand side of Figure 8, the two dimensional sinusoidal signals are drawn. The right hand side is the weighted sum of the other two. The original one-dimensional sinusoidal signals which create the plaid pattern, are plotted in Figure 10 *top*, where $\mathbf{b}_1$, $\mathbf{c}^1$, $\mathbf{b}_2$, $\mathbf{c}^2$ are plotted in order. This training data has been artificially contaminated with 50% outliers (see Figure 9*b*), where outliers are generated by uniformly sampling positions in the matrix and replacing the values with zero mean Gaussian noise having the variance of the signal.

The first row of Figure 10 shows the *true* factorization of the matrix (without outliers) into the two original sinusoids that generate the outer products. The term $\mathbf{b}_1$ in Figure 10*a* corresponds to the first column of matrix $\mathbf{B}$ while $\mathbf{c}^1$ corresponds to the first row of the coefficient matrix $\mathbf{C}$.

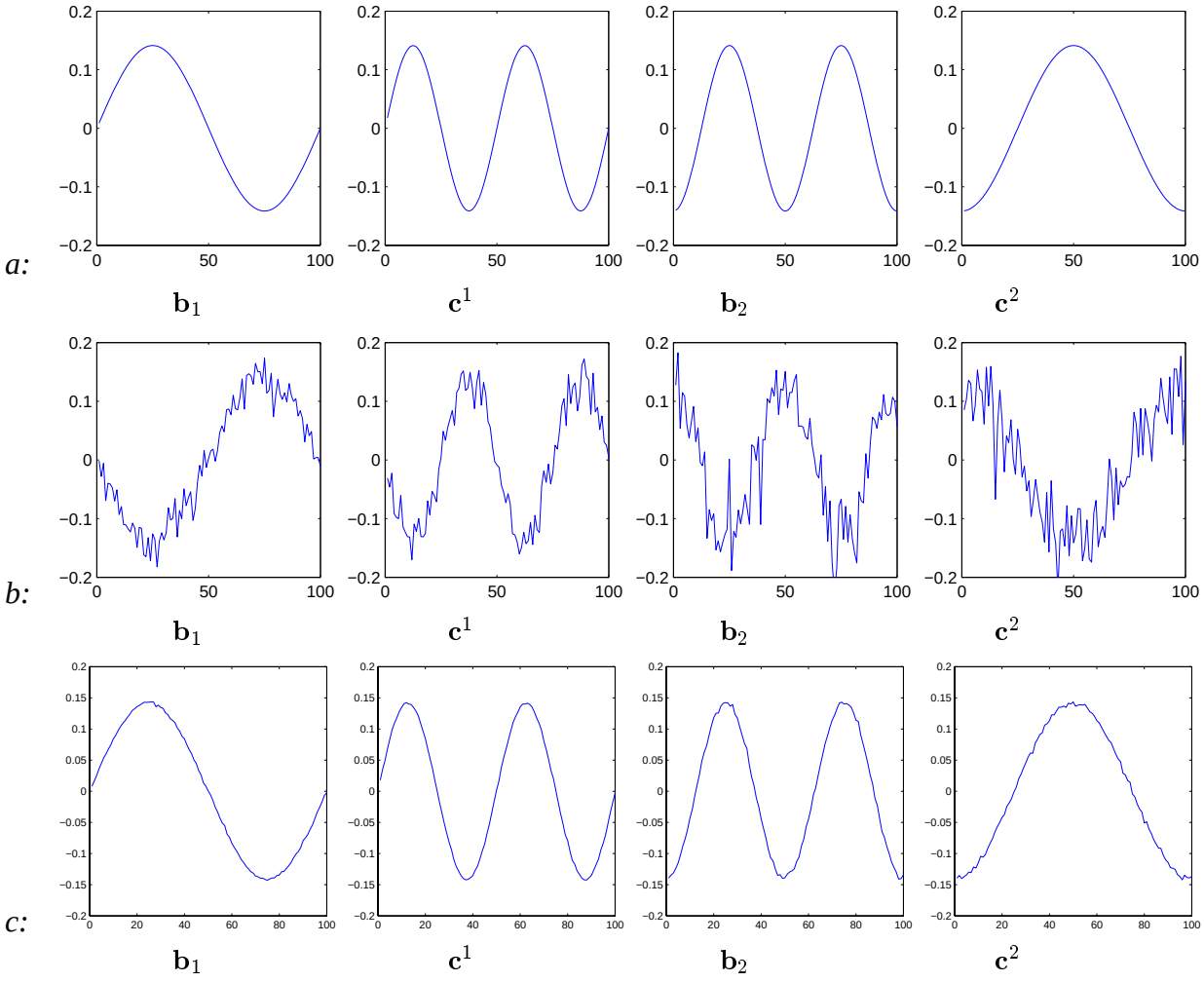


Figure 10: Robust SVD. *a.* Factorization of non-contaminated data. *b.* Bases and coefficients learned using traditional SVD. *c.* Bases and coefficients learned using Robust SVD.

Similarly $\mathbf{b}_2$ and $\mathbf{c}^2$ correspond to the second mode.

Figure 10*b* shows the least squares solution with the standard version of the SVD. Similarly, Figure 9*c* shows the full reconstructed matrix using the SVD result. As can be observed, due to the effects of outliers the solution is very noisy. In contrast, Figure 10*c* and Figure 9*d* show the robust solution to the factorization problem. Observe how the results achieved by the RSVD are closer to the original data than those obtained with SVD. The error between the original noiseless matrix ($\mathbf{D}$) and the reconstructed matrix ($\mathbf{BC}$) (i.e. $||\mathbf{D} - \mathbf{BC}||_F$) is 25.61 for standard SVD and 1.16 for RSVD. The original matrix $\mathbf{D}$, is produced from 2 sinusoidal signals and the first two eigenvalues of this matrix are 50.00 and 25.00 respectively. The eigenvalues recovered by RSVD are 49.27 and

24.60, whereas standard SVD, in contrast, spreads out the energy over all the eigenvalues with the first two having much lower values of 26.60 and 14.35.

5 Experiments

To illustrate the range of applications of Robust Subspace Learning (RSL), we consider two problems of current interest in computer vision. The first involves learning a "background" appearance model for use in person detection and tracking [52]. More generally, RSL can be applied to any other eigen-image learning problem. We also consider the problem of computing structure from the motion of tracked feature points. We show how both of these problems benefit from a robust formulation that can reject intra-sample outliers.

5.1 Learning a subspace for illumination

The behavior of RPCA is illustrated with a collection of 520 images (120×160) gathered from a static camera over two days. The first column in Figure 11 and 12, shows example training images; in addition to changes in the illumination of the static background, 40% of the images contain people in various locations. While the people often pass through the view of the camera quickly, they sometimes remain relatively still over multiple frames. We applied standard PCA and RPCA to the training data to build a background model that captures the illumination variation and that can be used to detect and track people [52].

The second column of Figure 11 and Figure 12 shows the result of reconstructing each of the illustrated training images using the PCA basis (with 20 basis vectors). The presence of people in the scene affects the recovered illumination of the background and results in "ghostly" images where the people are poorly reconstructed.

The third column shows the reconstruction obtained with 20 RPCA basis vectors. RPCA is able to capture the illumination changes while ignoring the people. In the fourth column, the outliers are plotted in white. Observe that the outliers primarily correspond to people, specular reflections, and graylevel changes due to the motion of the trees in the background. The last column plots the final weights, $\mathbf{W}^*$, for each image. Bright areas correspond to high weights (inliers) while black areas correspond to outliers. These weight images illustrate the "influence" that individual pixels

have on the recovered bases.

The RSL method does a better job of accounting for the illumination variation in the scene and provides a basis for person detection. The algorithm takes approximately eight hours on a 900 MHz Pentium III in Matlab (although a rough approximation of the basis takes around half hour). This is approximately an order of magnitude slower than SVD and is one of the disadvantages of the energy minimization formulation. The robust method is beneficial in situations where robustness is important, the dimensionality is relatively low, or learning can be performed off-line.

Figure 13*a* plots the value of the robust energy function versus the number of iterations of the core algorithm. Because the robust energy function depends on σ, in order to verify that at every iteration it decreases monotonically, we plot the energy function once σ has achieved its final, annealed, value. Figure 13*b* shows the convergence of the algorithm in terms of the principal angle between two consecutive subspaces $\mathbf{B}^n$ and $\mathbf{B}^{n+1}$ [28]. Since the energy function is being minimized, it decreases monotonically while the principal angle does not necessarily do so.

Figure 14 shows the mean (upper left) and the standard PCA bases learned here. Observe that the first few principal components appear to capture the major variations in illumination but that the effects of the people (outliers) appear in many of the bases as bright or dark regions. For comparison, Figure 15 shows the mean and bases recovered using RPCA. Note that the "patches" corresponding to the people do not appear.

5.2 Structure From Motion (SFM)

Recovering 3D shape and motion from feature correspondences across multiple views is a well known, and well studied, problem in computer vision. Here, we provide a brief overview of the factorization approach to structure from motion (SFM); for more details the reader is referred to [1, 36, 49, 54, 66]. The presentation here follows closely that of Irani and Anandan [36]. They address the problem of factoring shape and motion when there is measurable uncertainty in the locations of the features.

Given a set of n feature points of a rigid object tracked across d frames with coordinates $\{(x_{pi}, y_{pi}) |\ p = 1, ..., d, i = 1, ..., n\}$, the points can be stacked into the *measurement* matrix

$\mathbf{D} \in \Re^{2d \times n}$:

$$\mathbf{D} = \left[\begin{array}{c} \mathbf{X} \\ \mathbf{Y} \end{array} \right]_{2d \times n}. \tag{28}$$

At each time instant, we compute the mean of the feature points (i.e. the "center" of the object) and subtract the mean from the x, y feature locations. This defines a model of shape relative to the object's coordinate frame. The rows of the matrices $\mathbf{X}$ and $\mathbf{Y}$ contain these relative object coordinates at a single time instant.

It has been shown that, for an affine camera (i.e orthographic, weak-perspective, or paraperspective) [49, 54, 66], when there is no noise, the rank of $\mathbf{D}$ is 3 or less. Under these conditions, the matrix $\mathbf{D}$ can be factored into the product of a structure matrix $\mathbf{S} \in \Re^{3 \times n}$ and a motion matrix $\mathbf{M} \in \Re^{2d \times 3}$; that is, $\mathbf{D} = \mathbf{MS}$. The matrix $\mathbf{M}$ recovers the rotation of the object with respect to an arbitrary coordinate frame (e.g. the first frame) while the matrix $\mathbf{S}$ encodes the relative 3D positions (x, y, z) for each feature in the reconstructed object.

If there are errors (e.g. due to occlusion, missing data, or noise) the matrix $\mathbf{D}$ is no longer rank-3. A similar problem is posed by the presence of extra feature points corresponding to other independently moving objects in the scene. The problem of structure from motion with multiple moving objects is a difficult one (see [43] for a solution based on probabilistic mixture models). The robust formulation here is similar in spirit to work on robustly estimating multiple parameterized motions in the optical flow community [5].

When there are no outliers such as those above, a least-squares approximation can be found by minimizing $||\mathbf{D} - \mathbf{MS}||^2$ for any unitary invariant norm. In this case, we are implicitly assuming an isotropic noise model of the error (or it is optimal in this case); that is, the error for each feature p at each time instant i, $e_{pi} = d_{pi} - \sum_{j=1}^{3} m_{pj} s_{ji}$, is distributed as an isotropic Gaussian, $e_{pi} \sim N(0, \sigma^2)\ \forall p, i$. Under these assumptions, we can compute the SVD factorization of the measurement matrix $\mathbf{D} = \mathbf{U\Lambda V}^T$. Setting all but the largest three singular values to zero gives a matrix $\hat{\mathbf{\Lambda}}$. The best rank-3 approximation of $\mathbf{D}$ is then $\hat{\mathbf{D}} = \mathbf{U}\hat{\mathbf{\Lambda}}\mathbf{V}^T$. The matrices $\hat{\mathbf{M}} = \mathbf{U}\hat{\mathbf{\Lambda}}^{\frac{1}{2}}$ and $\hat{\mathbf{S}} = \hat{\mathbf{\Sigma}}^{\frac{1}{2}}\mathbf{V}^T$, provide a least squares estimate of motion and shape up to an affine transformation [66].

To deal with outliers, we apply the robust SVD techniques developed in this paper. In the SFM problem some additional constraints have to be taken into account in the robust factorization.

When either coordinate of a feature point (x or y) is an outlier, we would like to treat the entire point as an outlier; this implies some coupling between elements of $\mathbf{D}$ (the same thing happens in the case of optical flow or color images). In order to incorporate this additional constraint into our algorithm, we simply modify the robust ρ-function to depend on a vector valued input rather than a scalar:

$$\rho_2(\mathbf{x}, \sigma_p) = \frac{\mathbf{x}^T\mathbf{x}}{\mathbf{x}^T\mathbf{x} + \sigma_p^2}.$$

Let $\mathbf{r}_{ip} = [(d_{ip} - \mu x_p - \sum_{j=1}^{k} b_{pj}c_{ji}) \;\; (d_{(i+d)p} - \mu y_p - \sum_{j=1}^{k} b_{pj}c_{ji})]^T$, then the robust energy function becomes

$$E_{rpca2}(\mathbf{B}, \mathbf{C}) = \sum_{i=1}^{n}\sum_{p=1}^{d} \rho_2(\mathbf{r}_{ip}, \sigma_p). \tag{29}$$

The algorithm is basically the same as the one explained above and, in the interest of space, we will develop only the weighted least squares approach here. Given $\boldsymbol{\sigma}$ and some initial parameters, the error is computed as $\tilde{\mathbf{E}} = \mathbf{D} - \mathbf{MS} = \begin{bmatrix} \tilde{\mathbf{E}}_x \\ \tilde{\mathbf{E}}_y \end{bmatrix}$. Once $\tilde{\mathbf{E}}$ is computed, we define the joint error $\mathbf{E}_{joint} = \sqrt{\tilde{\mathbf{E}}_x{}^2 + \tilde{\mathbf{E}}_y{}^2} \in \Re^{d\times n}$, and every matrix element ip contains the error for each residual $\mathbf{r}_{ip}$. As before we can define a weight matrix $\mathbf{W} = \begin{bmatrix} \tilde{\mathbf{W}}_x \\ \tilde{\mathbf{W}}_y \end{bmatrix} \in \Re^{2d\times n}$, were $w_{pi} = w_{p(i+d)} = \frac{2\sigma_p^2}{(\mathbf{r}_{ip}^T\mathbf{r}_{ip} + \sigma_p^2)^2}$, so $\tilde{\mathbf{W}}_y = \tilde{\mathbf{W}}_x \in \Re^{d\times n}$. Once we have $\mathbf{W}$, the algorithm will alternate between solving (14), (15), (16), recomputing the error $\tilde{\mathbf{E}}$, and calculating the weight matrix $\mathbf{W}$.

Note there is some similarity in motivation and approach with the work of Morris and Kanade [49] and Irani and Anandan [36]. Irani and Anandan perform a covariance-weighted SVD that minimizes the Mahalanobis distance in feature space. They assume that the covariance can be factored. Morris and Kanade allow a general covariance matrix but do not provide a robust formulation.

In this section we report results of our experimental evaluation of the robust factorization algorithm and compare the results with traditional SVD. Following Irani and Anandan, we use similar synthetic data to analyze the performance of the algorithm. Figure 16 shows three frames of the original synthetic 3D data of a cube. The actual feature points are located at the intersections of the grid lines which are drawn for visualization purposes only. The cube undergoes rotational motion about the z axis. Figure 16 shows the orthographic projection in which 10% of the samples have been contained with outliers (the crosses). The outliers are synthetically generated from a uniform distribution in x and y coordinates and are different for each frame. These synthetic outliers simu-

late the problem of mismatches between points caused by failures of a feature tracker. The results for multiple independent motions are similar.

Figure 17*a* shows the standard SVD reconstruction of the shape animated with the recovered motion. As can be observed due to the outlying data, the estimation of the shape of the cube is very noisy. Figure 17*b* plots the solution obtained with the robust SVD method proposed in this paper, which produces much more accurate results. The error in the shape estimatet, $||\mathbf{S} - \hat{\mathbf{S}}||_F$, is 17.2314 and 1.3938 for traditional least squares and robust SVD respectively. Additionally the error in the motion, $||\mathbf{M} - \hat{\mathbf{M}}||_F$, is 0.4177 for traditional least squares and 0.0060 in the robust case.

Another issue of practical interest in the SFM computation involves missing data (e.g. when feature points do not appear in all the views). In the SFM problem, the missing points are typically known, and the weights can be set to zero for these points. With these weights fixed, the robust estimation can be performed as above.

6 Discussion and Related Methods

In this section, we explore other possible applications and extensions of RSL to computer vision problems. De la Torre and Black [19] proposed Robust Parameterized Component Analysis, a technique to robustly learn a subspace invariant to geometric transformations (useful when there is misalignment between the training images). Also, De la Torre and Black [17] have proposed Dynamic Coupled Component Analysis to robustly learn temporal and spatial dependencies between two or more high dimensional training sets. However, there exist many other subspace reletated problems which can benefit from a robust formulation. In the interest of space, we simply point out possible applications of the ideas developed in this paper. The robust formulation of many of these problems can proceed similarly to what is done here, though further research is needed.

6.1 Multi-linear Models

There exist problems in vision and signal processing that are best modeled by the interaction of multiple factors. One example is the work of Tenenbaum and Freeman on factoring style and content [64] using a bilinear model. There are numerous extensions to this idea in vision such

as modeling facial appearances as a linear combination of illumination, expression, and identity. Other multi-linear models include tensorial approaches to structure from motion [33] or Independent Component Analysis (ICA) [12]. Tensor factorization can be seen as a generalization of PCA to more than two dimensions. However there is no unique extension of PCA to multi-linear models; see for example [13, 37, 40]. If one views tensor factorization methods in terms of the minimization of an energy function, then the robust subspace learning methods developed here can be applied to multi-linear models in a relatively straightforward way.

6.2 Weighted Subspace Analysis

Weighted Subspace Analysis (WSA) provides a formalism for learning linear models when the data is weighted by known weights. Note that Equations (15) and (16) can be used to perform WSA. Recent work has used this approach for constructing appearance models of 3D articulated human figures from 2D image views [62]. WSA provides a formalism for constructing subspaces with missing data [61] and weighting data with different ranges (e.g. when constructing Active Appearance Models [15] where the shape and the graylevel pixels have different variance). This idea can be used for computing structure from motion when some measure of certainty of the tracked feature points is available [36, 49]. When the weights are separable, GSVD provides an efficient method for taking this pixel-weighted uncertainty into account. In the more general case, the method proposed here is straightforward to apply. Additionally, RSL provides a framework for on-line PCA/SVD computation as new data becomes available.

6.3 Minor Component Analysis (MCA)

Another obvious extension of this work is to the robust estimation of the subspace spanned by the smallest eigenvalues [51]. MCA is a useful technique for solving Total Least Squares problems [68]. The formulation of the robust optimization method, however, is not clear and further research is needed.

6.4 Regularized Component Analysis

In many situations it can be useful to find subspaces with spatial coherence between the bases. For instance, a subspace which captures illumination variations is likely to be composed of the sum of

smooth patterns or bases. In this case, we try to recover a smooth eigenspace in which the bases vary smoothly by minimizing $\sum_{i=1}^{n} ||\mathbf{d}_i - \mathbf{B}\mathbf{c}_i||_2^2 + \lambda \sum_{j=1}^{k} \mathbf{b}_j^T \mathbf{H} \mathbf{b}_j$ where $\mathbf{H}$ is a symmetric positive definite sparse matrix. Another application of regularization involves adding spatial coherence to outliers [63] if we expect them to correspond to coherent spatial structures.

6.5 The Eigenvalue Problem

Finding the Eigenvalues of a positive definite matrix $\mathbf{\Gamma} \in \Re^{d \times d}$ is important for many problems in applied mathematics [29, 53]. A possible future application of the work presented here is to robustify the symmetric eigenvalue problem by relating the eigenvalue problem to a robust energy minimization problem. While the energy minimization approach may make this extension feasible, it will come at some cost in terms of computation. Finding the subspace spanned by the largest eigenvectors can be achieved minimizing $E_{eig}(\mathbf{U}, \Lambda) = ||\mathbf{\Gamma} - \mathbf{U}\Lambda\mathbf{U}^T||_F$. It is easy to show that any saddle point of the energy function ($\frac{\partial E}{\partial \mathbf{U}} = 0$) is related to finding the solution of the eigenvalue problem $\mathbf{\Gamma U} = \mathbf{U}\Lambda$. Introducing an intra-sample outlier into E_{eig} could be useful when the matrix $\mathbf{\Gamma}$ contains outliers. Also note that if $\mathbf{\Gamma}$ is a covariance matrix and naturally expands as the sum of outer products, we can directly use the RPCA method proposed in this paper.

More generally, there are several other problems in computer vision (e.g. Linear discriminant Analysis [23, 44] (LDA) and segmentation [60]) that are based on the generalized eigenvalue problem. While formulating these applications using an intra-sample outlier process is relatively straightforward, the solution of the resulting robust generalized eigenvalue problem remains unclear.

7 Discussion

While the examples throughout the paper illustrate the benefits of the method, it is worth considering when the algorithm may give unwanted results. Consider, for example, a face database that contains a small fraction of the subjects wearing glasses. In this case, the pixels corresponding to the glasses are likely to be treated as outliers by RPCA. Hence, the learned basis set will not contain these pixels, and it will be impossible to reconstruct images of people wearing glasses. Whether or not this is desirable behavior will depend on the application.

In such a situation, people with or without glasses can be considered as two different classes of objects and it might be more appropriate to robustly learn multiple linear subspaces corresponding to the different classes. By detecting outliers, robust techniques may prove useful for identifying such training sets that contain significant subsets that are not well modeled by the majority of the data and should be separated and represented independently. This is one of the classic advantages of robust techniques for data analysis.

Another issue to take into account, is the fact that a training set can contain both intra-sample outliers and sample outliers. In order to solve such a problem, one should introduce both a sample outlier and intra-sample outlier in Equation (10). However, the learning rules would be complicated to derive. Another approach, would be a hierarchical one, where first the sample outliers are detected and removed from the training set. After that, we can apply the method proposed in this paper for removing the intra-sample outliers. In order to implement an efficient sample RPCA, using the same Iterative Reweighted Least Squares idea, given some initial weights, one can compute iteratively a weighted covariance matrix or if the data is very high dimensional perform the GSVD [31]. With this first estimation of the eigenvectors, one can calculate the error and computing the weights again and so on until convergence. Also a more practical approach would discard as sample outliers those samples which have more intra-sample outliers than a certain threshold.

8 Conclusions

We have presented a method for robust subspace learning that can be used for automatic learning of linear models from data that may be contaminated by outliers. In particular, we have applied this formalism to the problems of principal component analysis and singular value decomposition. The approach extends previous work in the vision community by modeling outliers that typically occur at the pixel level. Furthermore, it extends work in the statistics community by connecting the explicit outlier formulation with robust M-estimation and by developing a fully automatic algorithm that is appropriate for high dimensional data such as images. The method has been tested on natural and synthetic images and shows improved tolerance to outliers when compared with other techniques.

We have illustrated the methods with examples from eigen-image modeling and structure from

motion. These are important problems in computer vision and robust approaches may help provide solutions in situations with realistic amounts of un-modeled noise. In general, the use of linear models in vision is widespread and increasing. We hope robust techniques like those proposed here will prove useful as linear models are used to represent more realistic data sets. Towards that end, a Matlab implementation of the method and the results can be downloaded from

`http://www.salleURL.edu/~ftorre`

Acknowledgments

This work was supported by the DARPA HumanID Project (ONR contract N000140110886) and the National Science Foundation (ITR Program award #0113679). The first author was also partially supported by Catalonian Government grant 2000 BE I200132. MJB was also supported by a gift from the Xerox Foundation.

We thank Allan Jepson for many discussions on robust learning and PCA. We also thank Niko Troje for providing the face image database. Images from the Columbia database

http://www.cs.columbia.edu/CAVE/research/softlib/

were also used in the examples.

References

[1] P. Aguiar and J. Moura. Factorization as a rank 1 problem. In *Conference on Computer Vision and Pattern Recognition*, pages 178–184, 1999.

[2] P. Baldi and K. Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2:53–58, 1989.

[3] A. E. Beaton and J. W. Tukey. The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, 16(2):147–185, 1974.

[4] M. J. Black and P. Anandan. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, March 1996.

[5] M. J. Black and A. D. Jepson. Eigentracking: Robust matching and tracking of objects using view-based representation. *International Journal of Computer Vision*, 26(1):63–84, 1998.

[6] M. J. Black and A. Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *International Journal of Computer Vision*, 25(19):57–92, 1996.

[7] M. J. Black, G. Sapiro, D. Marimont, and D. Heeger. Robust anisotropic diffusion. *IEEE Transactions on Image Processing*, 7:421–432, 1998.

[8] M. J. Black, Y. Yaccob, A. Jepson, and D. J. Fleet. Learning parameterized models of image motion. In *Conference on Computer Vision and Pattern Recognition*, pages 561–567, 1997.

[9] A. Blake and M. Isard. *Active Contours*. Springer Verlag, 1998.

[10] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press series, Massachusetts, 1987.

[11] N. A. Campbell. Robust procedures in multivariate analysis I: Robust covariance estimation. *Applied Statistics*, 29(3):231–2137, January 1980.

[12] J. F. Cardoso. Independent component analysis, a survey of some algebraic methods. In *International Symposium Circuits and Systems,vol 2.*, pages 93–96, 1996.

[13] J. Carroll and J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization eckart-young decomposition. *Psychometrika*, 35:283–319, January 1970.

[14] A. Cichocki, R. Unbehauen, and E. Rummert. Robust learning algorithm for blind separation of signals. *Electronics Letters*, 30(17):1386–1387, 1993.

[15] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In *European Conference Computer Vision*, pages 484–498, 1998.

[16] C. Croux and P. Filzmoser. Robust factorization of a data matrix. In *COMPSTAT, Proceedings in Computational Statistics*, pages 245–249, 1998.

[17] F. de la Torre and M. J. Black. Dynamic coupled component analysis. In *Computer Vision and Pattern Recognition*, pages 643–650, 2001.

[18] F. de la Torre and M. J. Black. Robust principal component analysis for computer vision. In *International Conference on Computer Vision*, pages 362–369, 2001.

[19] F. de la Torre and M. J. Black. Robust parameterized component analysis: Theory and applications to 2d facial modeling. In *European Conference on Computer Vision*, pages 653–669, 2002.

[20] K. I. Diamantaras. *Principal Component Neural Networks (Therory and Applications).* John Wiley & Sons, 1996.

[21] C. Eckardt and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936.

[22] B. S. Everitt. *An Introduction to Latent Variable Models.* London: Chapman and Hall, 1984.

[23] K. Fukunaga. *Introduction to Statistical Pattern Recognition, Second Edition.* Academic Press.Boston, MA, 1990.

[24] K. R. Gabriel and C. L. Odoroff. Resistant lower rank approximation of matrices. In *Data Analysis and Informatics, III*, pages 23–30, 1984.

[25] K. R. Gabriel and S. Zamir. Lower rank approximation of matrices by least squares with any choice of weights. *Technometrics, Vol. 21, pp.*, 21:489–498, 1979.

[26] D. Geiger and R. Pereira. The outlier process. In *IEEE Workshop on Neural Networks for Signal Proc.*, pages 61–69, 1991.

[27] S. Geman and D. McClure. Statistical methods for tomographic image reconstruction. *Bulletin of the International Statistical Institute*, LII:4:5, 1987.

[28] G. Golub and C. F Van Loan. *Matrix Computations*. 2nd ed. The Johns Hopkins University Press, 1989.

[29] G. H. Golub and H. A. van der Vorst. Eigenvalue computation in the 20th century. *Journal of Computational and Applied Mathematics*, 123:35–65, 2000.

[30] M. J. Greenacre. *Theory and Applications of Correspondence Analysis*. Academic Press, London, 1984.

[31] M. J. Greenacre. Correspondence analysis of multivariate categorical data by weighted least squares. *Biometrika*, 75:457–467, 1988.

[32] F. Hampel, E. Ronchetti, P. Rousseeuw, and W. Stahel. *Robust Statistics: The Approach Based on Influence Functions*. Wiley, New York., 1986.

[33] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press., 2000.

[34] P. W. Holland and R. E. Welsch. Robust regression using iteratively reweighted least-squares. *Communications in Statistics*, (A6):813–827, 1977.

[35] P. J. Huber. *Robust Statistics*. New York ; Wiley, 1981.

[36] M. Irani and P. Anandan. Factorization with uncertainty. In *European Conference on Computer Vision*, pages 539–553, 2000.

[37] H. Neudecker J. R. Magnus. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley, 1999.

[38] I. T. Jolliffe. *Principal Component Analysis*. New York: Springer-Verlag, 1986.

[39] J. Karhunen and J. Joutsensalo. Generalizations of principal component analysis, optimization problems, and neural networks. *Neural Networks*, 4(8):549–562, 1995.

[40] P. Kroonenberg and J. de Leeuw. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45:69–97, 1980.

[41] S. H. Lai. Robust image alignment under partial occlusion and spatially varying illumination change. *Computer Vision and Image Understanding*, 78:84–98, 2000.

[42] G. Li. Robust regression. In D. C. Hoaglin, F. Mosteller, and J. W. Tukey, editors, *Exploring Data, Tables, Trends and Shapes*. John Wiley & Sons, 1985.

[43] J. MacLean, A. Jepson, and R. Frecker. Recovery of egomotion and segmentation of indepedent object motion using the EM-algorithm. In *British Machine Vision Conference*, pages 175–184, Leeds, UK, 1994.

[44] K. Mardia, J. Kent, and J. Bibby. *Multivariate Analysis*. Academic Press, London, 1979.

[45] P. Meer, D. Mintz, D. Kim, and A. Rosenfeld. Robust regression methods in computer vision: A review. *International Journal of Computer Vision*, 6:59–70, 1991.

[46] P. Meer, C. Stewart, and D. Tyler (Eds.). Special issue on robust statistics. *Computer Vision and Image Understanding*, 78(1), 2000.

[47] L. Mirsky. Symmetric gauge functions and unitarily invariant norms. *Quart. J. Marth. Oxford*, 11:50–59, 1960.

[48] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *Pattern Analysis and Machine Intelligence*, 19(7):137–143, July 1997.

[49] D. Morris and T. Kanade. A unified factorization algorithm for points, line segments and planes with uncertainty models. In *International Conference on Computer Vision*, pages 696–702, 1998.

[50] H. Murase and S. K. Nayar. Visual learning and recognition of 3D objects from appearance. *International Journal of Computer vision*, 1(14):5–24, 1995.

[51] E. Oja. A simplified neuron model as principal component analyzer. *Journal of Mathematical Biology*, 15:267–273, 1982.

[52] N. Oliver, B. Rosario, and A. Pentland. A Bayesian computer vision system for modeling human interactions. In H. I. Christensen, editor, *Int. Conf. Computer on Vision Systems, ICVS*, volume 1542 of *LNCS-Series*, pages 255–272, Gran Canaria, Spain, January 1999. Springer-Verlag.

[53] B. N. Parlett. *The Symmetric Eigenvalue Problem*. Prentice-Hall, Englewood Cliffs, NJ., 1980.

[54] C. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. In *International Conference on Computer Vision*, pages 97–108, 1994.

[55] R. P. N. Rao. An optimal estimation approach to visual perception and learning. *Vision Research*, 39(11):1963–1989, april 1999.

[56] P. J. Rousseeuw and A. M. Leroy. *Robust Regression and Outlier Detection*. John Wiley and Sons, 1987.

[57] S. Roweis. EM algorithms for PCA and SPCA. In *Neural Information Processing Systems*, pages 626–632, 1997.

[58] F. H. Ruymagaart. A robust principal component analysis. *Journal of Multivariate Analysis*, 11:485–497, 1981.

[59] T. D. Sanger. Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural Networks*, 2:459–473, November 1989.

[60] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), August 2000.

[61] H. Shum, K. Ikeuchi, and R. Reddy. Principal component analysis with missing data and its application to polyhedral object modeling. *Pattern Analysis and Machine Intelligence*, 17(9):855–867, 1995.

[62] H. Sidenbladh, F. de la Torre, and M. J. Black. A framework for modeling the appearance of 3D articulated figures. In *Face and Gesture Recognition*, pages 368–375, 2000.

[63] D. Skoaj, H. Bischof, and A. Leonardis. A robust PCA algorithm for building representations from panoramic images. In *European Conference Computer Vision*, pages 761–775, 2002.

[64] J. B. Tenenbaum and W. T. Freeman. Separating style and context with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

[65] M. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society B*, 61:611–622, 1999.

[66] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *Int. Jorunal of Computer Vision.*, 9(2):137–154, 1992.

[67] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal Cognitive Neuroscience*, 3(1):71–86, 1991.

[68] S. Van Huffel and J. Vandewalle. *The Total Least Squares Problem: Computational Aspects and Analysis*. Society for Industrial and Applied Mathematics, Philadelphia, 1991.

[69] L. Xu. Least mean square error recosntruction for self-organizing nerual nets. *Neural Networks*, 6:627–648, 1993.

[70] L. Xu and A. Yuille. Robust principal component analysis by self-organizing rules based on statistical physics approach. *IEEE Transactions on Neural Networks*, 6(1):131–143, 1995.

[71] T. N. Yang and S. D. Wang. Robust algorithms for principal component analysis. *Pattern Recognition Letters*, 20(9):927–933, 1999.

[72] A.L. Yuille, D. Snow, R. Epstein, and P. Belhumeur. Determining generative models for objects under varying illumination: Shape and albedo from multiple images using svd and integrability. *International Journal of Computer Vision*, 35(3):203–222, 1999.

[73] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and vision Computing*, 15(1):59–76, 1996.

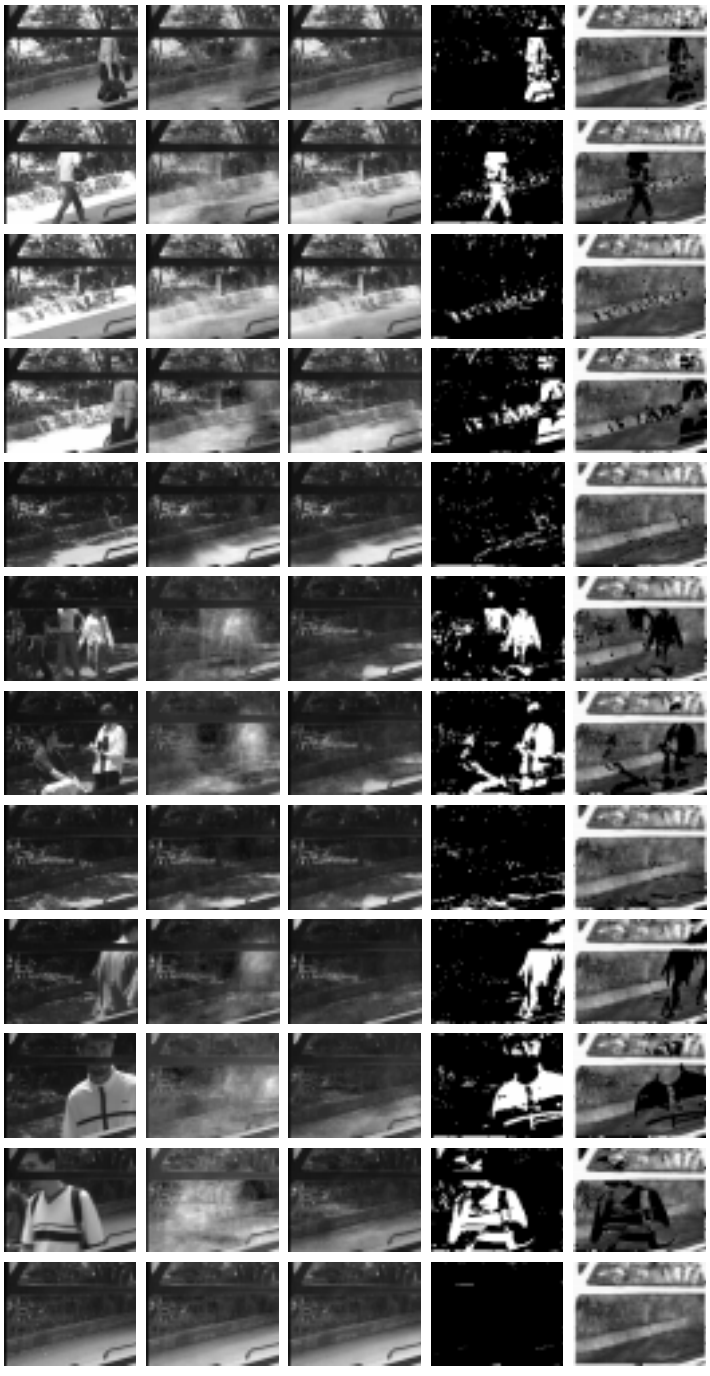


Figure 11: *(a)* Original Data. *(b)* PCA reconstruction. *(c)* RPCA reconstruction. *(d)* Outliers. *(e)* Weights.

Figure 12: *(a)* Original Data. *(b)* PCA reconstruction. *(c)* RPCA reconstruction. *(d)* Outliers. *(e)* Weights.

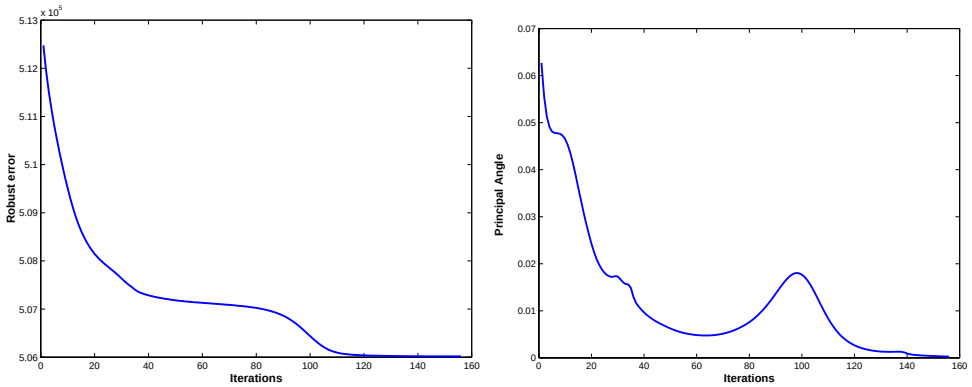


Figure 13: Convergence properties. *a.* Robust energy function vs. iterations *b.* Principal angle vs. iterations.

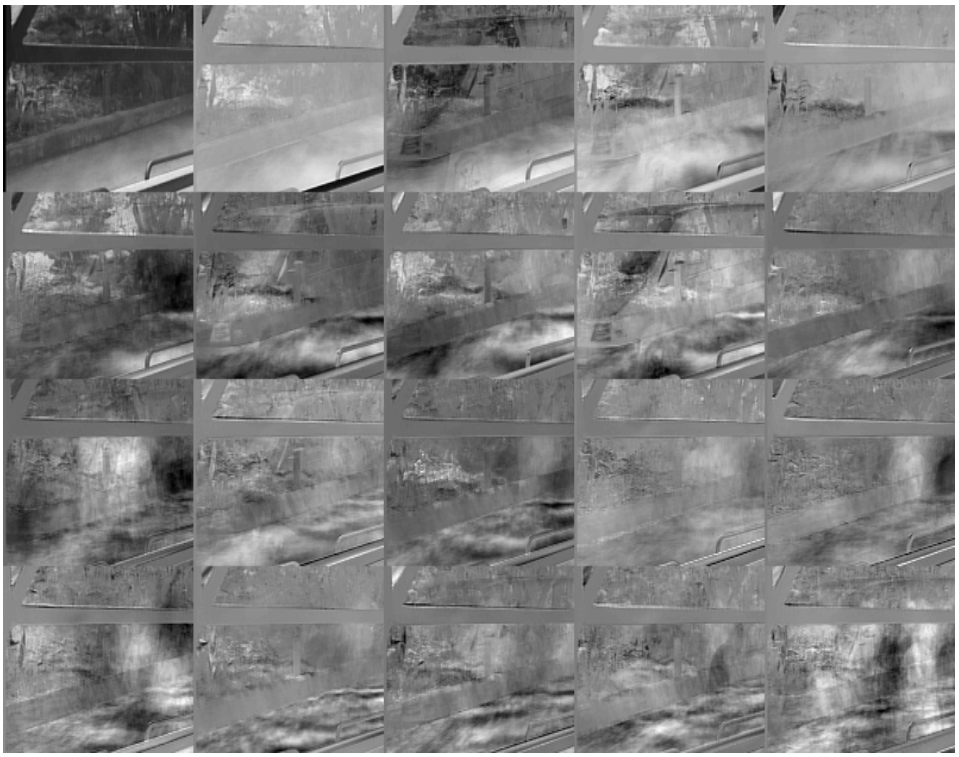


Figure 14: Standard PCA. The learned model using standard PCA. The mean image appears in the upper left followed by the bases from left to right, top to bottom. Notice the "patchy" effects of the people (outliers).

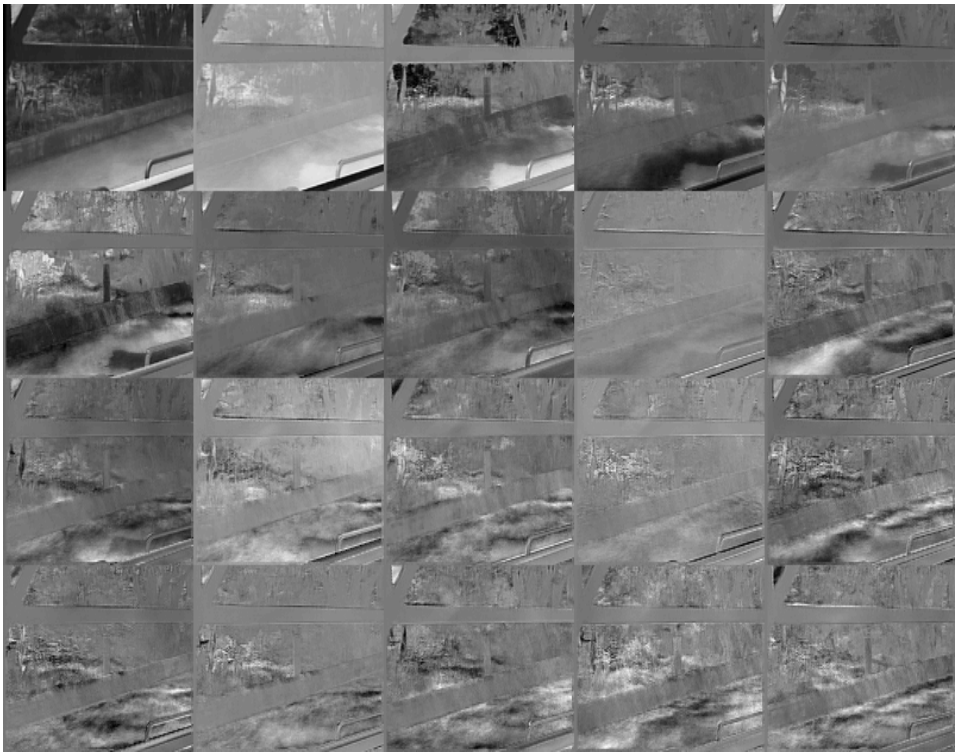


Figure 15: Robust PCA. Learned linear model using RPCA. Compared with Figure 14 the effect of outliers has been reduced and the bases primarily capture changes in illumination.

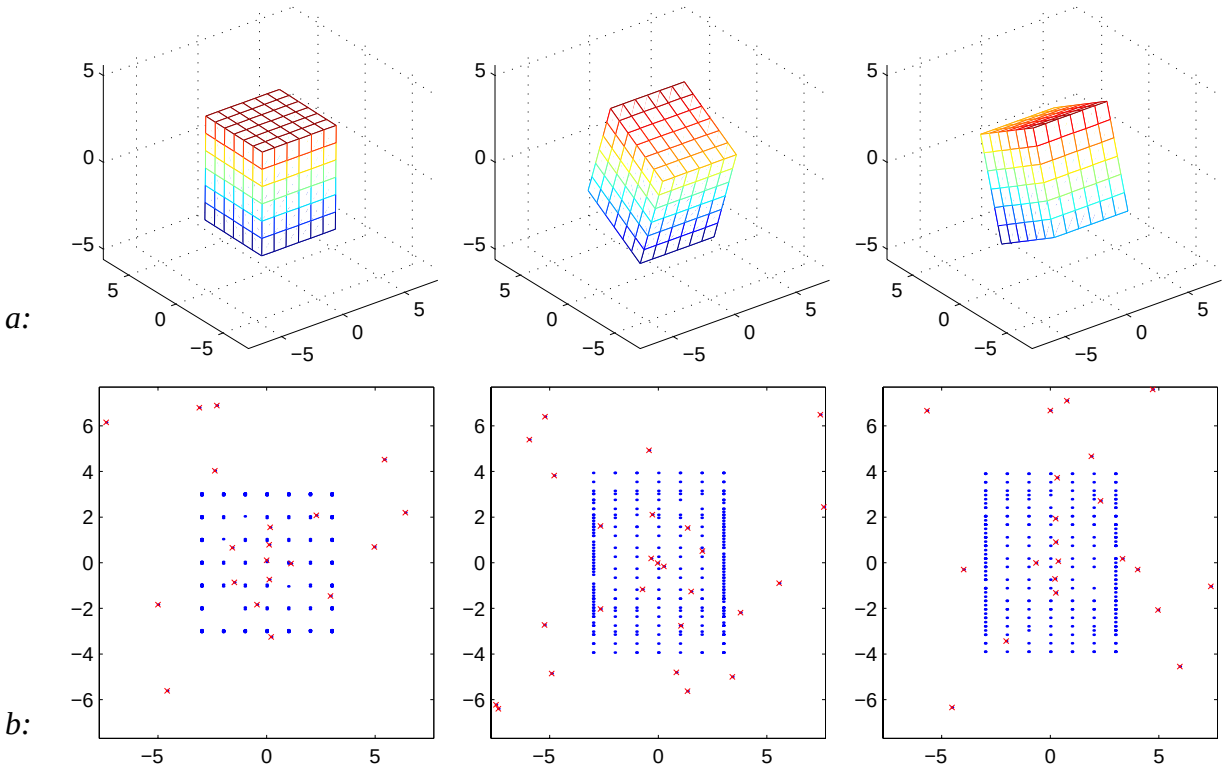


Figure 16: 3D data of a rotating cube. *a.* Three views of the cube as it rotates about the z axis. *b.* Orthographic projection of the feature points with the addition of outliers.

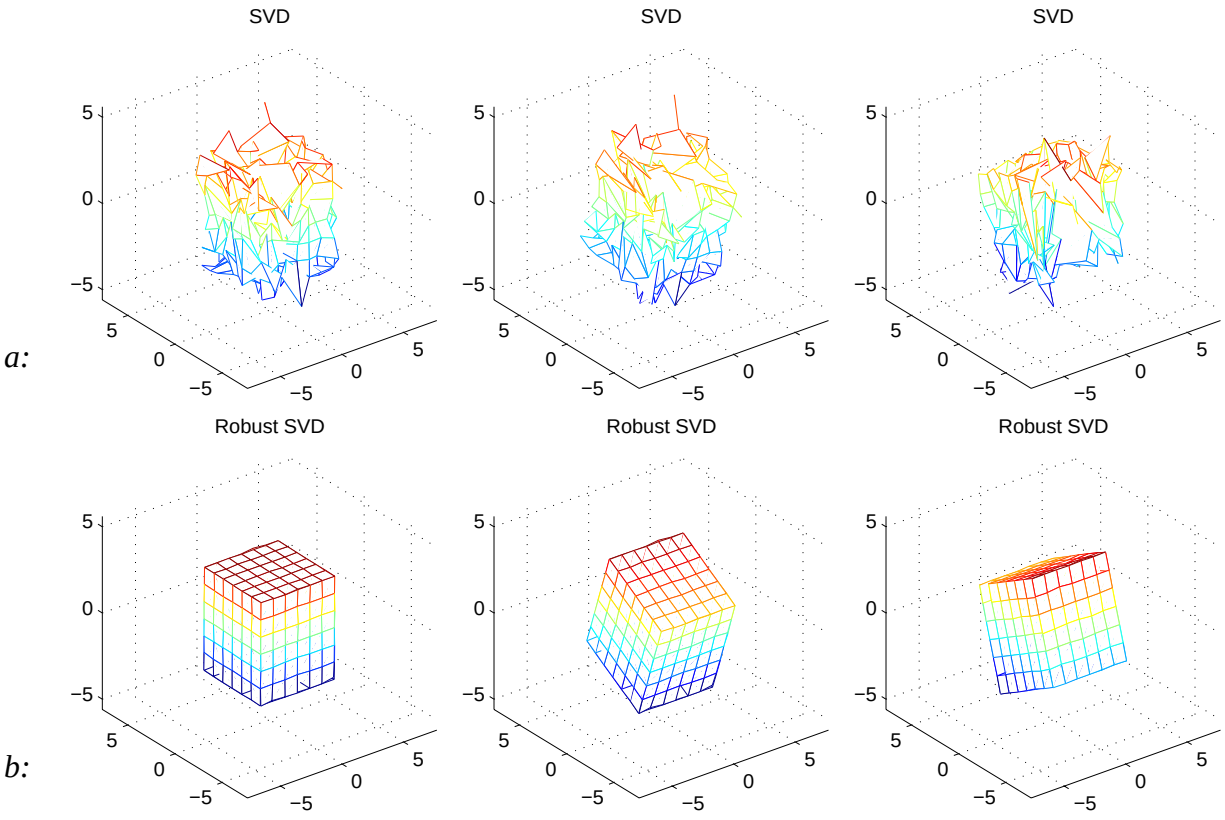


Figure 17: Reconstruction of shape and motion. The reconstructed shape of the cube is displayed with the view determined by the recovered motion. *a.* Standard Least Squares Factorization. *b.* Robust Factorization.