Report No. UIUCDCS-R-2006-2748 **UILU-ENG-2006-1788**

Regularized Locality Preserving Projections with Two-Dimensional Discretized Laplacian Smoothing

by

Deng Cai, Xiaofei He, and Jiawei Han

July 2006

Regularized Locality Preserving Projections with Two-Dimensional Discretized Laplacian Smoothing*

Deng Cai† Xiaofei He‡ Jiawei Han†

† Department of Computer Science, University of Illinois at Urbana-Champaign

‡ Yahoo! Research Labs

Abstract

A novel approach to linear dimensionality reduction is introduced that is based on Locality Preserving Projections (LPP) with a discretized Laplacian smoothing term. The choice of penalty allows us to incorporate prior information that some features may be correlated. For example, an $n_1 \times n_2$ image represented in the plane is intrinsically a matrix. The pixels spatially close to each other may be correlated. Even though we have $n_1 \times n_2$ pixels per image, this spatial correlation suggests the real number of freedom is far less. However, most of the previous methods consider an image as a vector in $\mathbb{R}^{n_1 \times n_2}$. They do not take advantage of the spatial correlation in the image, and the pixels are considered as independent pieces of information. In this paper, we introduce a Regularized LPP model using a Laplacian penalty to constrain the coefficients to be spatially smooth. By preserving the local geometrical structure of the image space, we can obtain a linear subspace which is optimal for image representation in the sense of local isometry. Recognition, clustering and retrieval can be then performed in the image subspace. Experimental results on face representation and recognition demonstrate the effectiveness of our method.

1 Introduction

Recently there are considerable interest in geometrically motivated approaches to visual analysis. The visual data like image and video is generally of very high dimensionality, ranging from several thousands to several hundreds of thousands. For example, a typical image of face is of size 32×32, resulting in a 1024-dimensional vector. However, the intrinsic degrees of freedom is far less. Various researchers (see [3], [5], [28], [30], [37]) have considered the case when the data lives on or close to a submanifold of the ambient space. One hopes then to estimate geometrical and topological properties of the submanifold from random points ("scattered data") lying on this unknown submanifold.

* The work was supported in part by the U.S. National Science Foundation NSF IIS-03-08215/IIS-05-13678. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

Previous works have demonstrated that the face recognition performance can be improved significantly in lower dimensional linear subspaces [2], [17], [21], [23], [31], [34]. Two of the most popular appearance-based face recognition methods include *Eigenface* [31] and *Fisherface* [17]. Eigenface is based on Principal Component Analysis (PCA) [10]. PCA projects the face images along the directions of maximal variances. It also aims to preserve the Euclidean distances between face images. For linearly embedded manifolds, PCA is guaranteed to discover the dimensionality of the manifold and produces a compact representation. Fisherface is based on Linear Discriminant Analysis (LDA) [10]. Unlike PCA which is unsupervised, LDA is supervised. When the class information is available, LDA can be used to find a linear subspace which is optimal for discrimination. Some extensions and variants of PCA and LDA have also been proposed, such as Penalized Discriminant Analysis [14], Kernel PCA [29], Kernel LDA [1], [35], etc.

Recently, the Locality Preserving Projection (LPP) algorithm is proposed to discover the local geometrical structure of the data space [16]. LPP is derived by finding the optimal linear approximations to the eigenfunctions of the Laplace Beltrami operator on the data manifold. The Laplace Beltrami operator takes the second order derivatives of the functions on the manifolds. It measures the smoothness of the functions. Therefore, LPP can discover the nonlinear manifold structure to some extent. LPP has demonstrated its effectiveness in face recognition. The basis functions obtained by LPP is generally referred to as *Laplacianfaces* [17].

Most of previous methods consider a face image as a high dimensional vector. They do not take advantage of the spatial correlation in the image, and the pixels are considered as independent pieces of information. However, a $n_1 \times n_2$ face image represented in the plane is intrinsically a matrix. Even though we have $n_1 \times n_2$ pixels per iamge, this spatial correlation suggests the real number of freedom is far less. In this paper, we introduce a Regularized LPP (RLPP) model using a Laplacian penalty to constrain the coefficients to be spatially smooth. Instead of considering the basis function as a $n_1 \times n_2$-dimensional vector, we consider it as a matrix, or a discrete function defined on a $n_1 \times n_2$ lattice. Thus, the discretized Laplacian can be applied to the basis functions to measure their smoothness along horizontal and vertical directions. The discretized Laplacian operator is a finite difference approximation to the second derivative operator, summed over all directions. The choice of Laplacian penalty allows us to incorporate the prior information that neighboring pixels are correlated.

Once we obtain compact representations of the images, classification and clustering can be performed in the lower dimensional subspace.

The points below highlight several aspects of the paper:

1. When the number-of-dimensions to sample-size ratio is too high, it is difficult for LPP to discover the intrinsic geometrical structure. Since the image data generally has a large number of dimensions (pixels), natural methods of regularization emerge.

2. Even if the sample size were sufficient to estimate the intrinsic geometrical structure, coefficients of spatially smooth features (pixels) tend to be spatially rough. Since we hope to interpret these coefficients, we would prefer smoother versions, especially if they do not compromise the fit.

3. The primary focus of this paper is on face images. However, our method can be naturally extended to higher order tensors, such as videos which are intrinsically the third order tensors. Our results may also be of interest to researchers in computer graphics who have considered the question of modeling the Bidirectional Texture Function (BTF) whose observational data is of six dimensions (i.e. sixth order tensor), two variables for surface location, two variables for view direction and two variables for illumination direction [22]. Researchers in computer vision, pattern recognition, molecular biology, information retrieval, and other areas where large amount of higher order tensor (rather than vector) based data are available may find some use of the algorithm and analysis of this paper.

The remainder of the paper is organized as follows. In Section 2, we provide a brief review of PCA, LDA and LPP. Section 3 describes the discretized Laplacian smoothing for image analysis. Section 4 introduces our proposed Regularized LPP algorithm. The extensive experimental results are presented in Section 5. Finally, we provide some concluding remarks and suggestions for future work in Section 6.

2 PCA, LDA and LPP

Suppose we have m $n_1 \times n_2$ face images. Let $\{\mathbf{x}_i\}_{i=1}^m \subset \mathbb{R}^n$ ($n = n_1 \times n_2$) denote their vector representations and $X = (\mathbf{x}_1, \cdots, \mathbf{x}_m)$.

2.1 PCA

PCA is a canonical linear dimensionality reduction algorithm. The basic idea of PCA is to project the data along the directions of maximal variances so that the reconstruction error can be minimized. Let $\mathbf{w}$ be the transformation vector and $y_i = \mathbf{a}^T\mathbf{x}_i$. Let $\boldsymbol{\mu} = \frac{1}{m}\sum \mathbf{x}_i$ and $\overline{y} = \frac{1}{m}\sum y_i$. The objective function of PCA is as follows:

$$\begin{aligned}
\mathbf{a}_{opt} &= \arg\max_{\mathbf{a}} \sum_{i=1}^{m} (y_i - \overline{y})^2 \\
&= \arg\max_{\mathbf{a}} \sum_{i=1}^{m} \mathbf{a}^T (\mathbf{x} - \boldsymbol{\mu}) (\mathbf{x} - \boldsymbol{\mu})^T \mathbf{a} \\
&= \arg\max_{\mathbf{a}} \mathbf{a}^T C \mathbf{a}
\end{aligned}$$

where $C = \frac{1}{m}\sum_{i=1}^{m} (\mathbf{x} - \boldsymbol{\mu}) (\mathbf{x} - \boldsymbol{\mu})^T$ is the data covariance matrix. The basis functions of PCA are the eigenvectors of the data covariance matrix associated with the largest eigenvalues.

2.2 LDA

Unlike PCA which is unsupervised, LDA is supervised. Suppose we have c classes and the i-th class have m_i samples, $m_1 + \cdots + m_c = m$. Let $\boldsymbol{\mu}_i$ be the sample mean vector of the i-th class. LDA aims to

maximize the ratio of between-class variance to the within-class variance thereby guaranteeing maximal separability. The objective function of LDA is as follows:

$$\max_{\mathbf{a}} \frac{\mathbf{a}^T S_b \mathbf{a}}{\mathbf{a}^T S_w \mathbf{a}} \tag{1}$$

where S_b is the *between-class scatter matrix* and S_w is the *within-class scatter matrix*. They are defined as follows:

$$S_b = \sum_{i=1}^{c} m_i \left(\boldsymbol{\mu}^i - \boldsymbol{\mu}\right) \left(\boldsymbol{\mu}^i - \boldsymbol{\mu}\right)^T$$

$$S_w = \sum_{i=1}^{c} \left(\sum_{j=1}^{m_i} \left(\mathbf{x}_j^i - \boldsymbol{\mu}^i\right) \left(\mathbf{x}_j^i - \boldsymbol{\mu}^i\right)^T \right)$$

where $\mathbf{x}_j^i$ is the j-th sample in the i-th class. Thus, the basis functions of LDA that maximize the objective function is given by the maximum eigenvalue solution to the generalized eigenvalue problem:

$$S_b \mathbf{a} = \lambda S_w \mathbf{a} \tag{2}$$

We can define the *total scatter matrix* as:

$$S_t = \sum_{i=1}^{m} \left(\mathbf{x}_i - \boldsymbol{\mu}\right) \left(\mathbf{x}_i - \boldsymbol{\mu}\right)^T = mC$$

It is easy to verify that $S_t = S_b + S_w$ [11], thus:

$$\begin{aligned}
& S_b \mathbf{a} = \lambda S_w \mathbf{a} \\
\Rightarrow \quad & (S_t - S_w)\mathbf{a} = \lambda S_w \mathbf{a} \\
\Rightarrow \quad & S_w \mathbf{a} = \frac{1}{1+\lambda} S_t \mathbf{a} \\
\Rightarrow \quad & S_b \mathbf{a} = \lambda (S_t - S_b)\mathbf{a} \\
\Rightarrow \quad & S_b \mathbf{a} = \frac{\lambda}{1+\lambda} S_t \mathbf{a}
\end{aligned}$$

Therefore, LDA can also be obtained by solving the following *minimum* eigenvalue problem:

$$S_w \mathbf{a} = \lambda S_t \mathbf{a} \tag{3}$$

or the following *maximum* eigenvalue problem:

$$S_b \mathbf{a} = \lambda S_t \mathbf{a} \tag{4}$$

2.3 LPP

Different from PCA and LDA which aim to discover the Euclidean structure, LPP aims to discover the local manifold structure. Given a similarity matrix S, the optimal projections can be obtained by solving the following minimization problem [16]:

$$\begin{aligned} \mathbf{a}_{opt} &= \arg\min_{\mathbf{a}} \sum_{ij} \left(\mathbf{a}^T\mathbf{x}_i - \mathbf{a}^T\mathbf{x}_j\right)^2 S_{ij} \\ &= \arg\min_{\mathbf{a}} \mathbf{a}^T X L X^T \mathbf{a} \end{aligned} \tag{5}$$

where $L = D - S$ is the *graph Laplacian* [9] and $D_{ii} = \sum_j S_{ij}$. The matrix D provides a natural measure on the data points. The bigger the value D_{ii} (corresponding to y_i) is, the more "important" is y_i. Therefore, we impose a constraint as follows:

$$\mathbf{y}^T D \mathbf{y} = 1 \Rightarrow \mathbf{a}^T X D X^T \mathbf{a} = 1,$$

where $\mathbf{y} = (y_1, \cdots, y_m)^T = X^T\mathbf{a}$.

Finally, the minimization problem reduces to finding:

$$\arg\min_{\mathbf{a}^T X D X^T \mathbf{a}=1} \mathbf{a}^T X L X^T \mathbf{a} \tag{6}$$

The transformation vector $\mathbf{a}$ that minimizes the objective function is given by the *minimum* eigenvalue solution to the generalized eigenvalue problem:

$$X L X^T \mathbf{a} = \lambda X D X^T \mathbf{a} \tag{7}$$

It is easy to see that:

$$\begin{aligned} & X L X^T \mathbf{a} = \lambda X D X^T \mathbf{a} \\ \Rightarrow \quad & X D X^T \mathbf{a} - X S X^T \mathbf{a} = \lambda X D X^T \mathbf{a} \\ \Rightarrow \quad & X S X^T \mathbf{a} = (1-\lambda) X D X^T \mathbf{a} \end{aligned}$$

Therefore, LPPs can also be obtained by solving the following *maximum* eigenvalue problem:

$$X S X^T \mathbf{a} = \lambda X D X^T \mathbf{a} \tag{8}$$

For the detailed derivation of LPP and the choices of S, please see [16].

2.4 Connections between PCA, LDA and LPP

In this subsection, we provide a discussion on connections between PCA, LDA and LPP. Our analysis is based on the different choices of graph structure that is inferred on the data points.

Connections between LDA and LPP

When the label information is available, it can be incorporated into the graph structure. We have the following proposition:

Proposition 1 *Suppose the data points have a zero mean vector. That is,* $\sum_i \boldsymbol{x}_i = 0$. *With the weight matrix defined as follows:*

$$S_{ij} = \begin{cases} 1/m_k, & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ both belong to the } k\text{-th class;} \\ 0, & \text{otherwise.} \end{cases}$$

LPP gives the same eigenvector solutions to LDA since $XSX^T = S_b$, $XLX^T = S_w$ *and* $XDX^T = S_t$.

Proof Please see [17] for the proof.

Proposition 1 shows that LDA tries to preserve the label information.

Proposition 2 *Suppose the data points have a zero mean vector. We have:*

$$rank(XSX^T) \leq c - 1$$

Proof Since the data points have a zero mean vector, $X\mathbf{e} = \mathbf{0}$ where $\mathbf{e} = (1, \cdots, 1)^T$ is a vector of all ones. Thus,

$$\begin{aligned} & X(\frac{1}{m}\mathbf{e}\mathbf{e}^T)X^T = 0 \\ \Rightarrow \quad & XSX^T = X(S - \frac{1}{m}\mathbf{e}\mathbf{e}^T)X^T \end{aligned}$$

Let

$$S - \frac{1}{m}\mathbf{e}\mathbf{e}^T = (\mathbf{s}_1^1, \cdots, \mathbf{s}_{m_1}^1, \mathbf{s}_1^2, \cdots, \mathbf{s}_{m_2}^2, \cdots, \mathbf{s}_1^c, \cdots, \mathbf{s}_{m_c}^c)$$

It is easy to see that

$$\mathbf{s}_1^i = \cdots = \mathbf{s}_{m_i}^i, \ i = 1, \cdots, c$$

and

$$\mathbf{s}_1^1 + \mathbf{s}_1^2 + \cdots + \mathbf{s}_1^c = 0$$

Therefore,

$$rank(XSX^T) \leq c - 1.$$

The singularity of XSX^T is generally referred to as *null space* problem in LDA [33], [36]. Our analysis indicates that the singularity of XSX^T results from the choices of the graph model. In this sense, a more reasonable S can be defined as follows:

$$S_{ij} = \begin{cases} \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\| \cdot \|\mathbf{x}_j\|}, & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ share the same label;} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

or

$$S_{ij} = \begin{cases} e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}}, & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ share the same label;} \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

Clear, the above choices of S no longer suffer from the null space problem.

Connections Between LPP and PCA

Let $\mathbf{x}$ and $\mathbf{y}$ be two independent random variables in the data space. We first define a ϵ *Covariance matrix* C_ϵ as follows.

Definition ϵ Covariance Matrix:

$$C_\epsilon = \frac{1}{2} E\Big[(\mathbf{x} - \mathbf{y}) (\mathbf{x} - \mathbf{y})^T \, | \|\mathbf{x} - \mathbf{y}\| < \epsilon \Big]$$

Let us recall the standard covariance matrix:

Definition Covariance Matrix: $C = E\Big[(\mathbf{x} - E[\mathbf{x}]) (\mathbf{x} - E[\mathbf{x}])^T \Big]$

We have the following propositions:

Proposition 3 $\lim_{\epsilon \to \infty} C_\epsilon = C$

Proposition 4 *With the weight matrix defined as follows:*

$$S_{ij} = \begin{cases} 1, & \textit{if } \|\mathbf{x}_i - \mathbf{x}_j\| < \epsilon \\ 0, & \textit{otherwise.} \end{cases} \tag{11}$$

We have: $\lim_{m \to \infty} \frac{1}{m^2 \beta} X L X^T = C_\epsilon$*, where* $\beta = Prob(\|\mathbf{x} - \mathbf{y}\| < \epsilon)$.

Please see [15] for the proofs of the above propositions. These propositions indicate that the matrix XLX^T provides a statistical estimation of the ϵ covariance matrix. Especially, when ϵ tends to infinity, XLX^T is just the sample covariance matrix.

Our analysis indicates that the choices of different graph structure play the central role of LPP. In some situations, one may incorporate prior information into the graph structure. For example, for web graph, one may connect two pages if there is a hyperlink between them [26]. In general, one can apply (9) for supervised learning and (11) for unsupervised learning.

3 Regularized LPP with Two-Dimensional Discretized Laplacian Smoothing

In this section, we describe how to apply Laplacian penalized functional to measure the smoothness of the basis vectors of the face space, which plays the key role in the regularized LPP with two-dimensional discretized laplacian smoothing algorithm. We begin with a general description of Laplacian smoothing.

3.1 Laplacian Smoothing

Let f be a function defined on a region of interest, $\Omega \subset \mathbb{R}^d$. The Laplacian operator $\mathcal{L}$ is defined as follows [19]:

$$\mathcal{L}f(\mathbf{t}) = \sum_{j=1}^{d} \frac{\partial^2 f}{\partial t_j^2} \tag{12}$$

The Laplacian penalty functional, denoted by $\mathcal{J}$, is defined by:

$$\mathcal{J}(f) = \int_{\Omega} \left[\mathcal{L}f\right]^2 d\mathbf{t} \tag{13}$$

Intuitively, $\mathcal{J}(f)$ measures the smoothness of the function f over the region Ω. In this paper, our primary interest is in image. An image is intrinsically a two-dimensional signal. Therefore, we take d to be 2 in the following.

3.2 Discretized Laplacian Smoothing

As we described previously, $n_1 \times n_2$ face images can be represented as vectors in $\mathbb{R}^n, n = n_1 \times n_2$. Let $\mathbf{a}_i \in \mathbb{R}^n$ be the basis vectors (transformation vectors) obtained by LPP. Without loss of generality, $\mathbf{a}_i$ can also be considered as functions defined on a $n_1 \times n_2$ lattice.

For a face image, the region of interest Ω is a two-dimensional rectangle, which for notational convenience we take to be $[0, 1]^2$. A lattice is defined on Ω as follows. Let $\mathbf{h} = (h_1, h_2)$ where $h_1 = 1/n_1$ and $h_2 = 1/n_2$. Ω_h consists of the set of two-dimensional vectors $\mathbf{t}_i = (t_{i_1}, t_{i_2})$ with $t_{i_j} = (i_j - 0.5) \cdot h_j$ for $1 \leq i_j \leq n_j$ and $1 \leq j \leq 2$. There are a total of $n = n_1 \times n_2$ grid points in this lattice. Let D_j be an $n_j \times n_j$ matrix that yields a discrete approximation to $\partial^2/\partial t_j^2$. Thus if $\mathbf{u} = (u(t_1), \cdots, u(t_{n_j}))$ is an n_j-dimensional vector which is a discretized version of a function $u(t)$, then D_j has the property that:

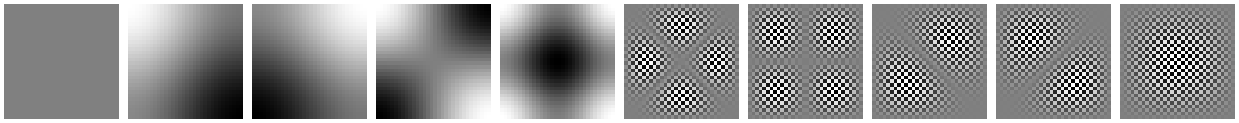
$$[D_j \mathbf{u}]_i \approx \frac{\partial^2 u(t_i)}{\partial t^2}$$

for $i = 1, \cdots, n_j$. There are many possible choices of D_j [6]. In this work, we apply the modified Neuman discretization [25]:

$$D_j = \frac{1}{h_j^2} \begin{pmatrix} -1 & 1 & & & & & & 0 \\ 1 & -2 & 1 & & & & & \\ & 1 & -2 & 1 & & & & \\ & & \cdot & \cdot & \cdot & & & \\ & & & 1 & -2 & 1 & & \\ & & & & 1 & -2 & 1 & \\ & & & & & 1 & -2 & 1 \\ 0 & & & & & & 1 & -1 \end{pmatrix}$$

Give D_j, a discrete approximation for two-dimensional Laplacian $\mathcal{L}$ is the $n \times n$ matrix:

$$\Delta = D_1 \otimes I_2 + I_1 \otimes D_2 \tag{14}$$



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 1: The first five (a∼e) and last five (f∼j) eigenvectors of $\Delta^T\Delta$. Δ is the discrete approximation for two-dimensional Laplacian as defined in Equation (14). Here $n_1 = n_2 = 32$ and thus Δ is a 1024×1024 matrix. All the eigenvectors are 1024-dimensional vectors and are displayed here as 32×32 images. The smoothness of eigenvectors can be measured by their corresponding eigenvalues. The smaller of the eigenvalue, the smoother of the eigenvector. The first five eigenvectors are spatially smooth while the last five eigenvectors are spatially rough.

where I_j is $n_j \times n_j$ identity matrix for $j = 1, 2$. $\otimes$ is the kronecker product defined below [18]:

Definition Let A be a $n \times n$ matrix and B be a $m \times m$ matrix. Then the kronecker product of A and B is the $mn \times mn$ block matrix

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{n1}B & \cdots & a_{nn}B \end{pmatrix}$$

For a $n_1 \times n_2$ dimensional vector $\mathbf{a}$, $\|\Delta\mathbf{a}\|$ provide a measure of smoothness of $\mathbf{a}$ on the $n_1 \times n_2$ lattice.

3.3 The Algorithm

Given a pre-defined graph structure with weight matrix S, the Regularized LPP is defined as the minimizer of

$$\sum_{ij} \left(\mathbf{a}^T\mathbf{x}_i - \mathbf{a}^T\mathbf{x}_j\right)^2 S_{ij} + \alpha\mathcal{J}(\mathbf{a}), \tag{15}$$

where $\mathcal{J}$ is the discretized Laplacian regularization functional:

$$\mathcal{J}(\mathbf{a}) = \|\Delta\mathbf{a}\|^2 = \mathbf{a}^T\Delta^T\Delta\mathbf{a}. \tag{16}$$

The parameter $\alpha > 0$ controls the smoothness of the estimator.

By simple algebraic formulations [16], we have:

$$\sum_{ij}(\mathbf{a}^T\mathbf{x}_i - \mathbf{a}^T\mathbf{x}_j)^2 S_{ij} = \mathbf{a}^T X L X^T \mathbf{a}.$$

With the same constraint as the standard LPP [16], finally the minimization problem reduces to finding:

$$\underset{\mathbf{a}^T X D X^T \mathbf{a}=1}{\arg\min_{\mathbf{a}}} \quad \mathbf{a}^T\left(XLX^T + \alpha\Delta^T\Delta\right)\mathbf{a}. \tag{17}$$

We will now switch to a Lagrangian formulation of the problem. The Lagrangian is as follows

$$\mathfrak{L} = \mathbf{a}^T \left(XLX^T + \alpha \Delta^T \Delta \right) \mathbf{a} - \lambda \mathbf{a}^T X D X^T \mathbf{a}. \tag{18}$$

Requiring that the gradient of $\mathfrak{L}$ vanish gives the following eigenvector problem:

$$\left(XLX^T + \alpha \Delta^T \Delta \right) \mathbf{a} = \lambda X D X^T \mathbf{a}. \tag{19}$$

It is easy to show that the matrices XLX^T, XDX^T and $\Delta^T \Delta$ are all symmetric and positive semi-definite. Since $\alpha > 0$, the matrix $XLX^T + \alpha \Delta^T \Delta$ is also symmetric and positive semi-definite. The vectors $\mathbf{a}_i$ $(i = 0, 1, \cdots, l-1)$ that minimize the objective function (17) are given by the minimum eigenvalue solutions to the above generalized eigenvalue problem.

4 Theoretical Analysis

In this section, we provide some theoretical analysis of the two-dimensional discrete Laplacian Δ as well as our regularized LPP algorithm.

Given a projection vector $\mathbf{a} \in \mathbb{R}^{n_1 \times n_2}$, its spatial smoothness can be measured as $\|\Delta \mathbf{a}\|$. To remove the impact of the norm of $\mathbf{a}$, we have the following definition:

Definition Let $\mathbf{a} \in \mathbb{R}^n, n = n_1 \times n_2$ be a projection vector. The *Discretized Laplacian Smoothing Function* $\mathcal{S}$ is defined as follows.

$$\mathcal{S}(\mathbf{a}) = \frac{\|\Delta \mathbf{a}\|^2}{\|\mathbf{a}\|^2} = \frac{\mathbf{a}^T \Delta^T \Delta \mathbf{a}}{\mathbf{a}^T \mathbf{a}} \tag{20}$$

$\mathcal{S}(\mathbf{a})$ measures the smoothness of the projection vector $\mathbf{a}$ over the $n_1 \times n_2$ lattice. The smaller $\mathcal{S}(\mathbf{a})$ is, the smoother $\mathbf{a}$ is.

It is easy to see that the "smoothest" $\mathbf{a}$ which minimizes $\mathcal{S}(\mathbf{a})$ is the eigenvector of $\Delta^T \Delta$ corresponding to the smallest eigenvalue. Figure 1 shows the first five and the last five eigenvectors of $\Delta^T \Delta$. The eigenvalues of $\Delta^T \Delta$ are exactly the values of $\mathcal{S}(\mathbf{a})$, where $\mathbf{a}$'s are the corresponding eigenvectors. As can be seen, the first five eigenvectors are spatially smoother than the last five eigenvectors. Particularly, the first eigenvector is a vector of all ones.

We have the following theorem:

Theorem 5 *The smallest eigenvalue of $\Delta^T \Delta$ is 0 and the corresponding eigenvector is $\boldsymbol{e} = (1, \cdots, 1)^T$, which is a vector of all ones.*

Proof $\Delta^T \Delta$ is positive semi-definite. All the eigenvalues of $\Delta^T \Delta$ are non-negative. It is sufficient to

show that $\mathbf{e}$ is the eigenvector of $\Delta^T\Delta$ corresponding to eigenvalue 0. We have:

$$\begin{aligned}
\Delta \cdot \mathbf{e} &= (D_1 \otimes I_2 + I_1 \otimes D_2) \cdot \mathbf{e} \\
&= (D_1 \otimes I_2) \cdot \mathbf{e} + (I_1 \otimes D_2) \cdot \mathbf{e} \\
&= \frac{1}{h_1^2} \begin{pmatrix} -I_2 & I_2 & & & 0 \\ I_2 & -2I_2 & I_2 & & \\ & \cdot & \cdot & \cdot & \\ & & I_2 & -2I_2 & I_2 \\ 0 & & & I_2 & -I_2 \end{pmatrix} \mathbf{e} + \begin{pmatrix} D_2 & & & & 0 \\ & D_2 & & & \\ & & \cdot & & \\ & & & D_2 & \\ 0 & & & & D_2 \end{pmatrix} \mathbf{e} \\
&= \mathbf{0} + \mathbf{0} = \mathbf{0}
\end{aligned}$$

Thus,

$$\Delta^T\Delta \cdot \mathbf{e} = \mathbf{0} = 0 \cdot \mathbf{e},$$

$\mathbf{e}$ is the eigenvector of $\Delta^T\Delta$ corresponding to eigenvalue 0.

Let λ_{LPP} and λ_{RLPP} be the smallest eigenvalues of equations (7) and (19), respectively,

$$\lambda_{LPP} = \min_{\substack{\mathbf{a} \\ \mathbf{a}^T X D X^T \mathbf{a} = 1}} \mathbf{a}^T X L X^T \mathbf{a} \tag{21}$$

and

$$\lambda_{RLPP} = \min_{\substack{\mathbf{a} \\ \mathbf{a}^T X D X^T \mathbf{a} = 1}} \mathbf{a}^T \left(X L X^T + \alpha \Delta^T \Delta \right) \mathbf{a} \tag{22}$$

Let $\mathbf{a}_{LPP}$ and $\mathbf{a}_{RLPP}$ be the corresponding eigenvectors. By definition (21), it is easy to see that:

$$\mathbf{a}_{LPP}^T X L X^T \mathbf{a}_{LPP} \leq \mathbf{a}_{RLPP}^T X L X^T \mathbf{a}_{RLPP}$$

This indicates that LPP has more locality preserving power than RLPP. As to the smoothness of the eigenvectors, we have the following theorem:

Theorem 6 $\|\Delta \mathbf{a}_{RLPP}\| \leq \|\Delta \mathbf{a}_{LPP}\|$

Proof By definition (21), we have:

$$\mathbf{a}_{LPP}^T X L X^T \mathbf{a}_{LPP} \leq \mathbf{a}_{RLPP}^T X L X^T \mathbf{a}_{RLPP}$$

By definition (22), we have:

$$\begin{aligned}
& \mathbf{a}_{RLPP}^T \big(X L X^T + \alpha \Delta^T \Delta \big) \mathbf{a}_{RLPP} \\
\leq\ & \mathbf{a}_{LPP}^T \big(X L X^T + \alpha \Delta^T \Delta \big) \mathbf{a}_{LPP} \\
\leq\ & \mathbf{a}_{RLPP}^T X L X^T \mathbf{a}_{RLPP} + \alpha \mathbf{a}_{LPP}^T \Delta^T \Delta \mathbf{a}_{LPP}
\end{aligned}$$

Subtracting $\mathbf{a}_{RLPP}^T X L X^T \mathbf{a}_{RLPP}$ from both sides and noticing that $\alpha > 0$, we get:

$$\|\Delta \mathbf{a}_{RLPP}\|^2 = \mathbf{a}_{RLPP}^T \Delta^T \Delta \mathbf{a}_{RLPP} \leq \mathbf{a}_{LPP}^T \Delta^T \Delta \mathbf{a}_{LPP} = \|\Delta \mathbf{a}_{LPP}\|^2$$

Theorem (6) indicates that the basis functions obtained by RLPP are spatially smoother than those obtained by LPP.

5 Learning Smooth Laplacianfaces for Representation

Based on Regularized LPP with two-dimensional discretized laplacian smoothing, we describe our *Smooth Laplacianfaces* method for face representation in this section. In recent years, there has been a growing interest in 2-d laplacian smoothing, as well as other higher order smoothing methods. These methods have been used in image de-noising [4], image reconstruction [7], [8] and image warping [13].

In the face analysis and recognition problem, one is confronted with the difficulty that the matrix XDX^T is sometimes singular. This stems from the fact that sometimes the number of images in the training set (m) is much smaller than the number of pixels in each image (n). In such a case, the rank of XDX^T is at most m, while XDX^T is a $n \times n$ matrix, which implies that XDX^T is singular. To overcome the complication of a singular XDX^T , we first project the image set to a PCA subspace so that the resulting matrix XDX^T is nonsingular. The algorithmic procedure of Smooth Laplacianfaces is formally stated below:

1. **PCA Projection**: We project the face images $\mathbf{x}_i$ into the PCA subspace by throwing away the components corresponding to zero eigenvalue. We denote the transformation matrix of PCA by W_{PCA}. By PCA projection, the extracted features are statistically uncorrelated and the rank of the new data matrix is equal to the number of features (dimensions). We denote as:
$$\widetilde{X} = W_{PCA}^T X \quad \text{and} \quad \widetilde{\Delta} = \Delta \cdot W_{PCA} \tag{23}$$

2. **Constructing the Adjacency Graph**: Let G denote a graph with n nodes. The i-th node corresponds to the face image $\mathbf{x}_i$. We put an edge between nodes i and j if $\mathbf{x}_i$ and $\mathbf{x}_j$ are "close", i.e. $\mathbf{x}_i$ is among k nearest neighbors of $\mathbf{x}_j$ or $\mathbf{x}_j$ is among k nearest neighbors of $\mathbf{x}_i$. Note that, if the class information is available, we simply put an edge between two data points belonging to the same class.

3. **Choosing the Weights**: If node i and j are connected, put
$$S_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}}$$
Otherwise, put $S_{ij} = 0$. The weight matrix S of graph G models the local structure of the face manifold. The justification of this weight can be traced back to [3].

4. **Eigenmap**: Compute the eigenvectors and eigenvalues for the generalized eigenvector problem:
$$\left(\widetilde{X}L\widetilde{X}^T + \alpha\widetilde{\Delta}^T\widetilde{\Delta}\right)\mathbf{a} = \lambda\widetilde{X}D\widetilde{X}^T\mathbf{a} \tag{24}$$
where D is a diagonal matrix whose entries are column (or row, since S is symmetric) sums of S, $D_{ii} = \sum_j S_{ji}$. $L = D - S$ is the Laplacian matrix [9].

Let $\mathbf{a}_0, \mathbf{a}_1, \cdots, \mathbf{a}_{l-1}$ be the solutions of (24), ordered according to their eigenvalues, $0 \leq \lambda_0 \leq \lambda_1 \leq \cdots \leq \lambda_{l-1}$. These eigenvalues are equal to or greater than zero because the matrix $\widetilde{X}L\widetilde{X}^T + \alpha\widetilde{\Delta}^T\widetilde{\Delta}$ is symmetric and positive semi-definite and $\widetilde{X}D\widetilde{X}^T$ is symmetric and positive definite. Thus, the embedding is as follows:

$$\mathbf{x} \rightarrow \mathbf{y} = W^T\mathbf{x} \tag{25}$$

$$W = W_{PCA}W_{RLPP} \tag{26}$$

$$W_{RLPP} = [\mathbf{a}_0, \mathbf{a}_1, \cdots, \mathbf{a}_{l-1}] \tag{27}$$

where $\mathbf{y}$ is a l-dimensional vector and W is the transformation matrix. This linear mapping not only preserves the manifold's estimated intrinsic geometry in a linear sense but also considers the spatial correlation of image pixels. The column vectors of W are the so-called *Smooth Laplacianfaces*.

6 Experimental Results

In this section, several experiments are carried out to show the effectiveness of our proposed Smooth Laplacianfaces method for face representation and recognition.

6.1 Face Representation Using Smooth Laplacianfaces

As we described previously, a face image can be represented as a point in image space. A typical image of size $n_1 \times n_2$ describes a point in $n_1 \times n_2$-dimensional image space. However, due to the unwanted variations resulting from changes in lighting, facial expression, and pose, the image space might not be an optimal space for visual representation.

In Section 5, we have discussed how to learn a spatially smooth locality preserving face subspace. The images of faces in the training set are used to learn such a subspace. The subspace is spanned by a set of eigenvectors of Eqn. (24), i.e., $\mathbf{a}_0, \mathbf{a}_1, \cdots, \mathbf{a}_{l-1}$. We can display the eigenvectors as special ghost-like images. These images may be called *Smooth Laplacianfaces* (S-Laplacianfaces). Using the Yale face database as the training set, we present the first 5 S-Laplacianfaces in Fig. (2), together with Eigenfaces, Fisherfaces and Laplacianfaces. Note that there is a parameter α which controls the smoothness in S-Laplacianfaces. Fig. (2) shows three groups S-Laplacianfaces with $\alpha = 0.5$, 5 and 50. For each face (eigenvector $\mathbf{a}$), we also calculated the $\|\Delta\mathbf{a}\|$. Since each eigenvector is normalized, $\|\Delta\mathbf{a}\|$ can measure the smoothness of $\mathbf{a}$ as we discussed in Section (4).

We can see that S-Laplacianfaces is smoother than Laplacianfaces. With bigger α, S-Laplacianfaces become much smoother. The Fisherfaces and Laplacianfaces are somehow similar to each other since they share similar graph structure as we described in Section 2. The Eigenfaces is the smoothest among all the faces. However, Eigenfaces do not encode discriminating information thus will not optimal for recognition. S-Laplacianfaces consider both the discriminating power and the spatial correlation among

$\|\Delta\mathbf{a}\| =$ 0.87 1.22 1.25 1.65 1.95

(a) Eigenfaces

$\|\Delta\mathbf{a}\| =$ 16.1 14.3 13.4 14.1 12.8

(b) Fisherfaces

$\|\Delta\mathbf{a}\| =$ 14.7 16.4 15.8 14.8 14.9

(c) Laplacianfaces

$\|\Delta\mathbf{a}\| =$ 10.7 10.8 11.8 11.6 12.4

(d) Smooth Laplacianfaces ($\alpha = 0.5$)

$\|\Delta\mathbf{a}\| =$ 7.56 7.19 7.91 7.45 8.27

(e) Smooth Laplacianfaces ($\alpha = 5$)

$\|\Delta\mathbf{a}\| =$ 3.89 3.82 4.39 4.29 4.26

(f) Smooth Laplacianfaces ($\alpha = 50$)

Figure 2: The first 5 Eigenfaces, Fisherfaces, Laplacianfaces, and Smooth Laplacianfaces calculated from the face images in the Yale database. For each face (eigenvector $\mathbf{a}$), we also calculated the $\|\Delta\mathbf{a}\|$. Since each eigenvector is normalized, $\|\Delta\mathbf{a}\|$ can measure the smoothness of $\mathbf{a}$ as we discussed in Section (4). S-Laplacianfaces is smoother than Laplacianfaces and Fisherfaces. With bigger α, S-Laplacianfaces become much smoother. It is also interesting to note that the eigenvectors of PCA corresponding to the largest eigenvalues are smoothest.

Figure 3: Sample face images from the Yale database. For each subject, there are 11 face images under different lighting conditions with facial expression.

the pixels on the face image, thus might achieve better performance on recognition. The recognition experiments in next section will demonstrate this.

6.2 Face Recognition Using Smooth Laplacianfaces

In this section, we investigate the performance of our proposed Smooth Laplacianface method for face recognition. The system performance is compared with the Eigenface (PCA) [32], Fisherface (LDA) [2], and Laplacianface (LPP) [17]. We use the same graph structures in the Laplacianface and our S-Laplacianface, which is built based on the label information.

Four face databases were used, i.e. Yale[1], ORL [2], Yale-B[3] and PIE (Pose, Illumination, and Expression)[4]. In all the experiments, preprocessing to locate the faces was applied. Original images were normalized (in scale and orientation) such that the two eyes were aligned at the same position. Then, the facial areas were cropped into the final images for matching. The size of each cropped image in all the experiments is 32×32 pixels, with 256 gray levels per pixel. Each image is represented by a $1,024$-dimensional vector in image space. Many pattern classifiers have been applied to face recognition, like nearest-neighbor classifier [31], Bayesian [24], and support vector machines [27], etc. In this work, we apply nearest neighbor classifier for its simplicity.

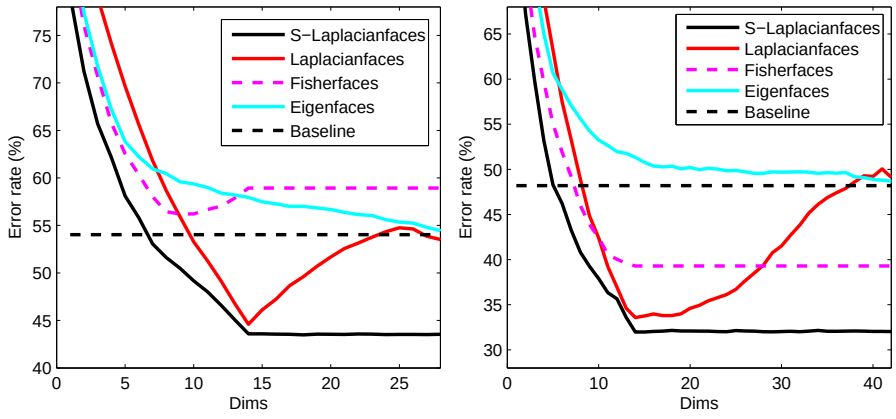
In short, the recognition process has three steps. First, we calculate the face subspace from the training set of face images; then the new face image to be identified is projected into d-dimensional subspace; finally, the new face image is identified by nearest neighbor classifier.

6.2.1 Yale Database

The Yale face database was constructed at the Yale Center for Computational Vision and Control. It contains 165 gray scale images of 15 individuals. The images demonstrate variations in lighting condition, facial expression (normal, happy, sad, sleepy, surprised, and wink). Fig. (3) shows the 11 images of one individual in Yale data base. A random subset with l $(= 2, 3, 4, 5)$ images per individual was taken with labels to form the training set, and the rest of the database was considered to be the testing set.

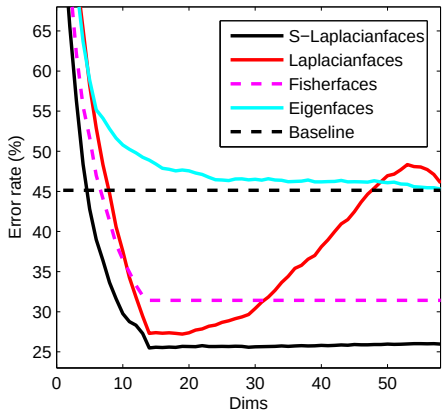
The training set is utilized to learn the subspace representation of the face manifold by using Eigenface,

[1] http://cvc.yale.edu/projects/yalefaces/yalefaces.html
[2] http://www.cl.cam.ac.uk/Research/DTG/attarchive/facesataglance.html
[3] http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html
[4] http://www.ri.cmu.edu/projects/project_418.html

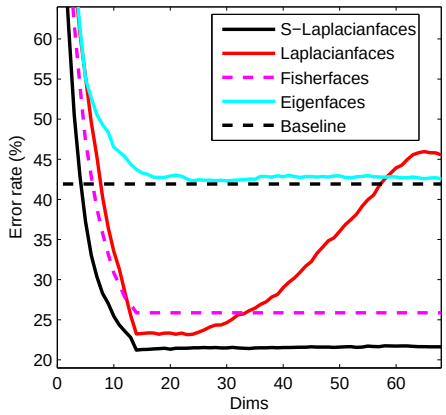


(a) 2 Train

(b) 3 Train



(c) 4 Train



(d) 5 Train

Figure 4: Error rate vs. dimensionality reduction on Yale database

Fisherface, Laplacianface and our algorithm. The testing images are projected into the face subspace in which recognition is then performed. For each given l, we average the results over 50 random splits. The cross validation in the training set was used to select the parameter α in our S-Laplacianface algorithm.

Figure 4 shows the plots of error rate versus dimensionality reduction for the Eigenface, Fisherface, Laplacianface, S-Laplacianface and baseline methods. For the baseline method, the recognition is simply performed in the original 1024-dimensional image space without any dimensionality reduction. Note that, the upper bound of the dimensionality of Fisherface is $c-1$ where c is the number of individuals [2]. As can be seen, the performance of the Eigenface, Fisherface, Laplacianface, and S-Laplacianface algorithms varies with the number of dimensions. We show the best results obtained by them in Table 1 and the corresponding face subspaces are called optimal face subspace for each method.

It is interesting to note that S-Laplacianface reaches the best performance almost always at $c-1$ dimensions. Table 1 lists the performance of S-Laplacianface at dimensionality $c-1$. This property shows that S-Laplacianface does not suffer from the problem of dimensionality estimation which is a crucial problem for most of the subspace learning based face recognition methods.

Table 1: Performance comparison on Yale database

Method	2 Train		3 Train		4 Train		5 Train	
	error	dim	error	dim	error	dim	error	dim
Baseline	54.0%	1024	48.2%	1024	45.1%	1024	41.9%	1024
Eigenfaces	54.0%	29	48.2%	44	45.1%	158	41.9%	74
Fisherfaces	56.2%	9	39.3%	14	31.4%	14	25.9%	14
Laplacianfaces	44.6%	14	33.6%	14	27.2%	19	23.1%	23
S-Laplacianfaces*	**43.5%**	18	**32.0%**	14	**25.5%**	14	**21.2%**	14
	43.6%	14	**32.0%**	14	**25.5%**	14	**21.2%**	14

* The first row of S-Laplacianfaces indicates the best performance as well as the optimal subspace dimension. The second row indicates the performance of S-Laplacianfaces at exactly $c-1$ dimension, c is the number of class.

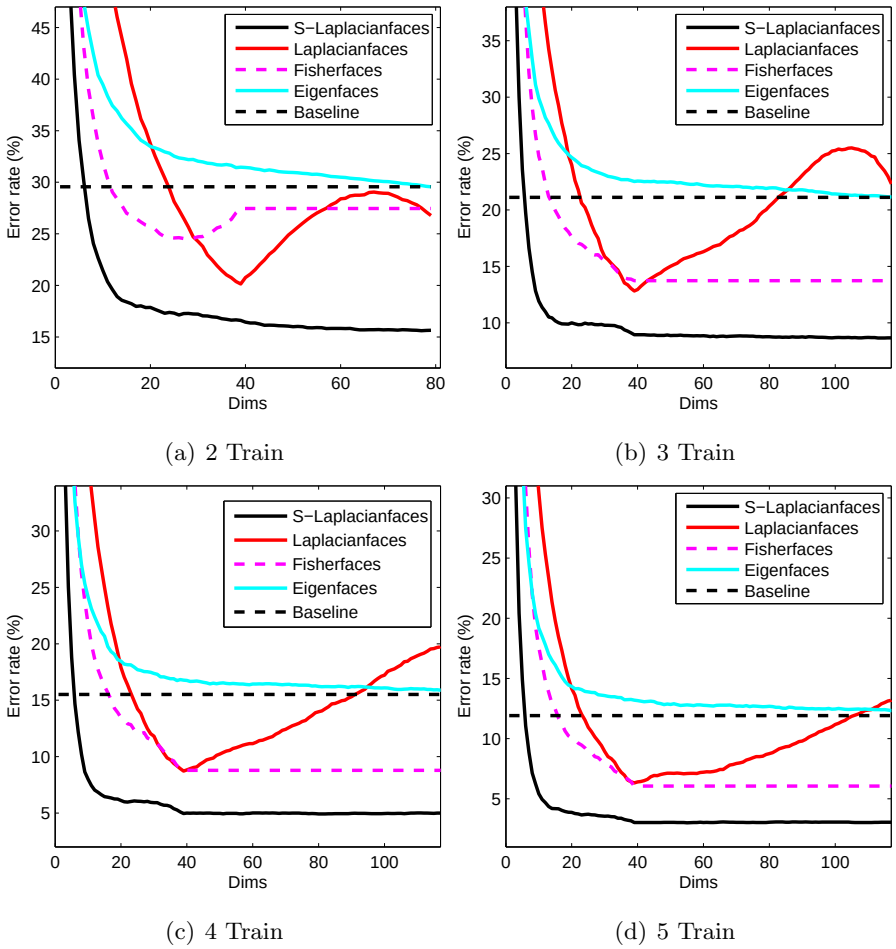
Figure 5: Sample face images from the ORL database. For each subject, there are 10 face images with different facial expression and details.

S-Laplacianface outperforms the other four methods with different numbers of training samples (2, 3, 4, 5) per individual. The Eigenface method performs the worst in most the cases. It does not obtain any improvement over the baseline method. The Laplacianface method perform slightly better than Fisherface method. It is worthwhile to note that in the cases where only two training samples are available, Fisherfaces method works even worse than baseline and Eigenfaces method. This result is consistent with the observation in [23] that Eigenface method can outperform Fisherface method when the training set is small. Moreover, in this case, the best performance of Fisherface is no longer obtained in a $c-1(=14)$ dimensional subspace, but a 9-dimensional subspace.

6.2.2 ORL Database

The ORL (Olivetti Research Laboratory) face database is used in this test. It consists of a total of 400 face images, of a total of 40 people (10 samples per person). The images were captured at different times and have different variations including expressions (open or closed eyes, smiling or non-smiling) and facial details (glasses or no glasses). The images were taken with a tolerance for some tilting and rotation of the face up to 20 degrees. 10 sample images of one individual are displayed in Figure 5. For each individual, $l(=2,3,4,5)$ images are randomly selected for training and the rest are used for testing.

The experimental design is the same as that in the last subsection. For each given l, we average the results over 50 random splits. Figure 6 shows the plots of error rate versus dimensionality reduction for the Eigenface, Fisherface, Laplacianface, S-Laplacianface and baseline methods. The best result obtained



(a) 2 Train (b) 3 Train

(c) 4 Train (d) 5 Train

Figure 6: Error rate vs. dimensionality reduction on ORL database

in the optimal subspace are shown in Table 2. It would be interesting to note that, when there are only two training samples for each individual, the best performance of Fisherface is no longer obtained in a $c - 1(= 39)$ dimensional subspace, but a 28-dimensional subspace.

As can be seen, our S-Laplacianface algorithm performs the best for all the cases and it reaches the best performance with $c - 1$ dimensions. The Fisherface and Laplacianface methods perform comparatively to each other, while the Eigenface method performs poorly.

6.2.3 Extended Yale Database B

The Extended Yale face database B contains 16128 images of 28 human subjects under 9 poses and 64 illumination conditions [12][20]. In this experiment, we choose the frontal pose and use all the images under different illumination, thus we get 64 images for each person. All the faces are manually aligned, cropped and resize to 32×32 pixels. 30 sample images of one individual are presented in Figure 7. For each individual, $l(= 5, 10, 20, 30)$ images are randomly selected for training and the rest are used for

Table 2: Performance comparison on ORL database

Method	2 Train		3 Train		4 Train		5 Train	
	error	dim	error	dim	error	dim	error	dim
Baseline	29.6%	1024	21.1%	1024	15.5%	1024	11.9%	1024
Eigenfaces	29.6%	79	21.1%	119	15.5%	158	11.9%	189
Fisherfaces	24.5%	28	13.7%	39	8.8%	39	6.1%	39
Laplacianfaces	20.1%	39	12.8%	39	8.7%	39	6.3%	39
S-Laplacianfaces*	**15.6%**	77	**8.7%**	113	**4.9%**	82	**3.0%**	39
	16.6%	39	**9.0%**	39	**5.0%**	39	**3.0%**	39

* The first row of S-Laplacianfaces indicates the best performance as well as the optimal subspace dimension. The second row indicates the performance of S-Laplacianfaces at exactly $c-1$ dimension, c is the number of class.

Figure 7: Sample face images from the extended Yale database B. For each subject, we use 64 frontal face images under varying illumination condition.

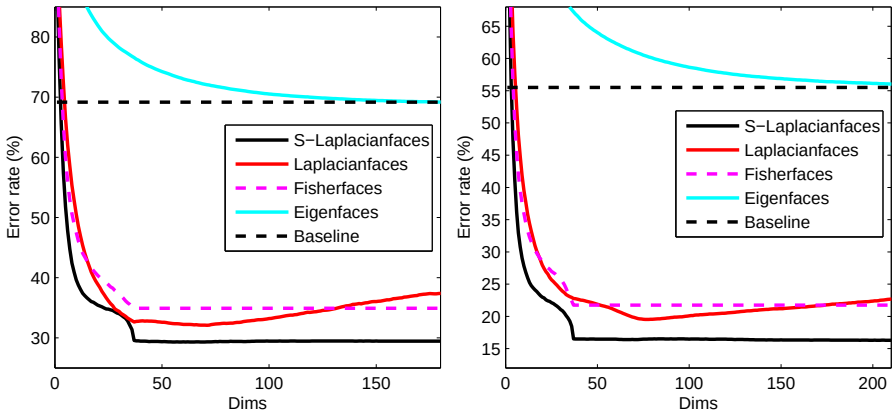
testing.

The experimental design is the same as that in the last subsection. For each given l, we average the results over 50 random splits. Table 3 shows the recognition results. As can be seen, our S-Laplacianface algorithm performs significantly better than the other algorithms. Fisherface and Laplacianface perform worse than S-Laplacianface, but much better than Eigenface. Figure 8 shows a plot of error rate versus dimensionality reduction. S-Laplacianface achieves a reasonably good performance with $c-1$ dimensions. There is no significant improvement if more dimensions are used.

6.2.4 PIE Database

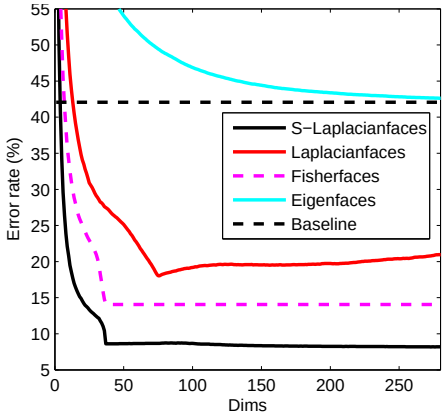
The CMU PIE face database contains 68 subjects with 41,368 face images as a whole. The face images were captured by 13 synchronized cameras and 21 flashes, under varying pose, illumination and expression. We choose the five near frontal poses (C05, C07, C09, C27, C29) and use all the images under different illuminations and expressions, thus we get 170 images for each individual. Figure 9 shows some of the faces with pose, illumination and expression variations. For each individual, $l(= 5, 10, 20, 30)$ images are randomly selected for training and the rest are used for testing.

The experimental design is the same as that in the last subsection. For each given l, we average the results over 50 random splits. Figure 10 shows the plots of error rate versus dimensionality reduction for

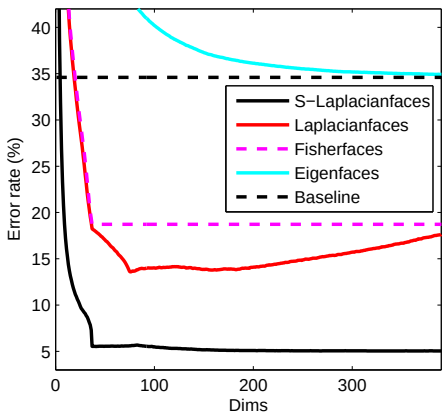


(a) 5 Train

(b) 10 Train



(c) 20 Train



(d) 30 Train

Figure 8: Error rate vs. dimensionality reduction on extended Yale database B

the Eigenface, Fisherface, Laplacianface, S-Laplacianface and baseline methods. The best results and the corresponding optimal face subspace for each method are shown in Table 4.

6.2.5 Discussion

Four experiments on Yale, ORL, Yale-B and PIE databases have been systematically performed. These experiments reveal a number of interesting points:

1. Five face recognition methods are compared, among which our S-Laplacianface algorithm performs the best. The Fisherface, Laplacianface and S-Laplacianface methods perform significantly better than the baseline method. This indicates that subspace learning is important for face recognition. The Eigenface method fails to gain improvement over the baseline method. This is probably due to the fact that the Eigenface method does not encode discriminating information.

2. All these algorithms can take advantage of more training samples, which is important to the real-world face recognition systems.

Table 3: Performance comparison on extended Yale database B

Method	5 Train		10 Train		20 Train		30 Train	
	error	dim	error	dim	error	dim	error	dim
Baseline	69.2%	1024	55.5%	1024	42.1%	1024	34.6%	1024
Eigenfaces	69.2%	189	55.5%	378	42.1%	616	34.6%	780
Fisherfaces	34.9%	37	21.7%	37	14.1%	37	18.7%	37
Laplacianfaces	32.1%	71	19.5%	76	18.0%	75	13.6%	76
S-Laplacianfaces*	**29.3%**	69	**16.2%**	280	**8.2%**	273	**5.0%**	387
	29.6%	37	**16.5%**	37	**8.6%**	37	**5.5%**	37

* The first row of S-Laplacianfaces indicates the best performance as well as the optimal subspace dimension. The second row indicates the performance of S-Laplacianfaces at exactly $c-1$ dimension, c is the number of class.

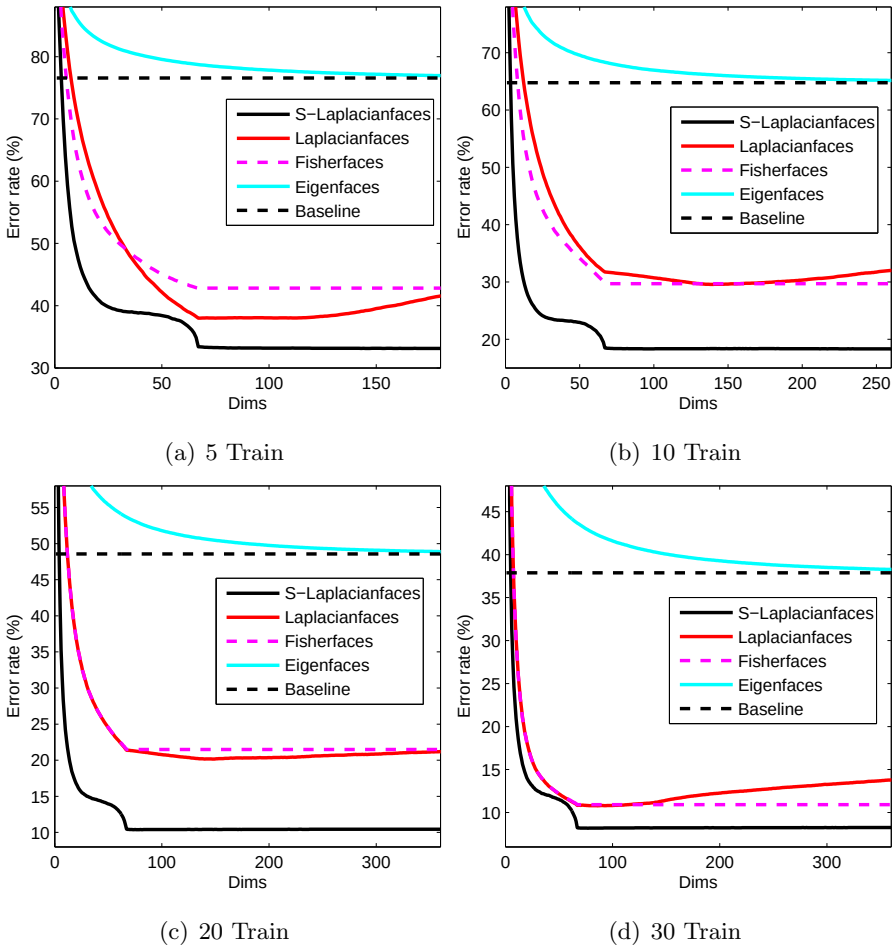
Figure 9: Sample face images from the CMU PIE database. For each subject, there are 170 near frontal face images under varying pose, illumination, and expression.

3. Comparing to the Laplacianface method, S-Laplacianface explicitly takes into account the spatial relationship between the pixels in an image. The use of spatial information significantly reduces the number of degrees of freedom. Therefore, S-Lapalcianface can have good performance even when there is only a small number of training samples available.

7 Conclusions

In this paper, we propose a new linear dimensionality reduction method called Regularized Locality Preserving Projections (RLPP). RLPP explicitly considers the spatial relationship between the pixels in images. By introducing a Laplacian penalized functional, the projection vectors obtained by RLPP can be smoother than those obtained by the ordinary LPP. This prior information significantly reduces the number of degrees of freedom, and hence RLPP can perform better than LPP when there is no sufficient training samples. We applied our RLPP method to face recognition and compared with Eigenface, Fisherface and Laplacianface methods on Yale, ORL, PIE, and Yale-B databases. Experimental results show that our method consistently outperforms the other methods.

The primary focus of this paper is on images which are two-dimensional signals. However, the analysis



(a) 5 Train (b) 10 Train

(c) 20 Train (d) 30 Train

Figure 10: Error rate vs. dimensionality reduction on PIE database

and algorithm presented here can also be naturally extended to higher dimensional signals. For example, a video can be considered as a three-dimensional signal, and thus a three-dimensional discretized Laplacian functional can be applied to video. Other examples include Bidirectional Texture Function (BTF) which is a six-dimensional signal. We are currently investigating the applicability of our algorithm for these problems.

References

[1] G. Baudat and F. Anouar. Generalized discriminant analysis using a kernel approach. *Neural Computation*, 12(10):2385–2404, 2000.

[2] Peter N. Belhumeur, J. P. Hepanha, and David J. Kriegman. Eigenfaces vs. fisherfaces: recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):711–720, 1997.

Table 4: Performance comparison on PIE database

Method	5 Train		10 Train		20 Train		30 Train	
	error	dim	error	dim	error	dim	error	dim
Baseline	76.6%	1024	64.8%	1024	48.6%	1024	37.9%	1024
Eigenfaces	76.6%	334	64.8%	654	48.6%	982	37.9%	1023
Fisherfaces	42.8%	67	29.7%	67	21.5%	67	10.9%	67
Laplacianfaces	38.0%	67	29.6%	139	20.2%	146	10.8%	86
S-Laplacianfaces*	**33.0%**	294	**18.1%**	280	**10.4%**	76	**8.2%**	77
	33.4%	67	**18.5%**	67	**10.5%**	67	**8.3%**	67

* The first row of S-Laplacianfaces indicates the best performance as well as the optimal subspace dimension. The second row indicates the performance of S-Laplacianfaces at exactly $c-1$ dimension, c is the number of class.

[3] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*, pages 585–591. MIT Press, Cambridge, MA, 2001.

[4] Mark Berman. Automated smoothing of image and other regularly spaced data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):460–468, 1994.

[5] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 16*, 2003.

[6] B. L. Buzbee, G. H. Golub, and C. W. Nielson. On direct methods for solving poisson's equations. *SIAM Journal on Numerical Analysis*, 7(4):627–656, Dec. 1970.

[7] Antonin Chambolle and Pierre-Louis Lions. Image recovery via total variation minimization and related problems. *Numerische Mathematik*, 76:167–188, 1997.

[8] Tony Chan, Antonio Marquina, and Pep Mulet. High-order total variation-based image restoration. *SIAM J. Sci. Comput.*, 22(2):503–516, 2000.

[9] Fan R. K. Chung. *Spectral Graph Theory*, volume 92 of *Regional Conference Series in Mathematics*. AMS, 1997.

[10] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, Hoboken, NJ, 2nd edition, 2000.

[11] Keinosuke Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, 2nd edition, 1990.

[12] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):643–660, 2001.

[13] C. A. Glasbey and K. V. Mardia. A review of image-warping methods. *Journal of Applied Statistics*, 25(2):155–171, 1997.

[14] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. New York: Springer-Verlag, 2001.

[15] Xiaofei He, Deng Cai, and Wanli Min. Statistical and computational analysis of lpp. In *Proceedings of the* 22^{nd} *International Conference on Machine Learning*, Bonn, Germany, 2005.

[16] Xiaofei He and Partha Niyogi. Locality preserving projections. In *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2003.

[17] Xiaofei He, Shuicheng Yan, Yuxiao Hu, Partha Niyogi, and Hong-Jiang Zhang. Face recognition using laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(3):328–340, 2005.

[18] Roger Horn and Charles Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

[19] Jurgen Jost. *Riemannian Geometry and Geometric Analysis*. Springer-Verlag, 2002.

[20] K.C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Transactions Pattern Analysis and Machine Intelligence*, 27(5):684–698, 2005.

[21] Z. Li, W. Liu, D. Lin, and X. Tang. Nonparametric subspace analysis for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2005.

[22] X. Liu, Y. Yu, and H.-Y. Shum. Synthesizing bidirectional texture functions for real-world surfaces. In *Proc. ACM SIGGRAPH*, pages 97–106, Los Angeles, CA, 2001.

[23] A. M. Martinez and A. C. Kak. PCA versus LDA. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2):228–233, 2001.

[24] B. Moghaddam. Principal manifolds and probabilistic subspaces for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6), 2002.

[25] Finbarr O'Sullivan. Discretized laplacian smoothing by fourier methods. *Journal of the American Statistical Association*, 86(415):634–642, Sep. 1991.

[26] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.

[27] P. J. Phillips. Support vector machines applied to face recognition. *Advances in Neural Information Processing Systems*, 11:803–809, 1998.

[28] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[29] B. Scholkopf, A. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[30] J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

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

[32] M. Turk and A. P. Pentland. Face recognition using eigenfaces. In *IEEE Conference on Computer Vision and Pattern Recognition*, Maui, Hawaii, 1991.

[33] X. Wang and X. Tang. Random sampling lda for face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2004.

[34] X. Wang and X. Tang. A unified framework for subspace face recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1222–1228, September 2004.

[35] Ming-Hsuan Yang. Kernel eigenfaces vs. kernel fisherfaces: Face recognition using kernel methods. In *Proc. of the fifth International Conference on Automatic Face and Gesture Recognition*, Washington, D. C., May 2002.

[36] H. Yu and J. Yang. A direct LDA algorithm for high dimensional data-with application to face recognition. *Pattern Recognition*, 34(10):2067–2070, 2001.

[37] Hongyuan Zha and Zhenyue Zhang. Isometric embedding and continuum isomap. In *Proc. of the twentieth Internation Conference on Machine Learning*, 2003.