Diffusion maps

Merzlikina Polina
Kravchenko Oleksandr

mentor:
senior researcher, PhD,
Sytnyk Dmytro

Semi-Supervised Learning on Riemannian Manifolds (2004) Mikhail Belkin, Partha Niyogi

Diffusion maps (2006) Ronald R.Coifman, Stéphane Lafon

Fundamental Limitations of Spectral Clustering (2007) Boaz Nadler, Meirav Galun

How it works?

X - data set
μ - distribution of the points on X

kernel $k: X \times X \to R$
k is symmetric: k(x,y) = k(y,x),
k is positivity preserving: k(x,y)>=0

Gaussian kernel (example):

$$k(x, y) = \exp\left(-\frac{||x - y||^2}{\epsilon}\right)$$

define the diffusion matrix L: $L_{i,j} = k(x_i, x_j)$

define the new kernel:

$$L_{i,j}^{(\alpha)} = k^{(\alpha)}(x_i, x_j) = \frac{L_{i,j}}{(d(x_i)d(x_j))^{\alpha}}$$

or equivalently:

$$L^{(\alpha)} = D^{-\alpha} L D^{-\alpha} \qquad D_{i,i} = \sum_j L_{i,j}.$$

Apply the graph Laplacian normalization to this new kernel:

$$M = (D^{(\alpha)})^{-1} L^{(\alpha)}$$

$$D_{i,i}^{(\alpha)} = \sum_j L_{i,j}^{(\alpha)}.$$

M properties:

- symmetric;
- positive defined;
- has a set of n real eigenvalues whose corresponding eigenvectors are the biorthogonal right and left eigenvectors.

$$M_{i,j}^{t} = \sum_{l} \lambda_{l}^{t} \psi_{l}(x_i) \phi_{l}(x_j)$$

left eigenvectors:

$$\psi \mathrm{M} = \lambda \psi$$

right eigenvectors:

$$\mathrm{M} \varphi = \lambda \varphi$$

Due to the spectrum decay of the eigenvalues, only a few terms are necessary to achieve a given relative accuracy in this sum.

raw data:

$X \in \mathbb{R}^n$

new coords

$$\Psi_t(x) = (\lambda_1^t \psi_1(x), \lambda_2^t \psi_2(x), \ldots, \lambda_k^t \psi_k(x))$$

$$\Psi_t(x) \in \mathbb{R}^k$$

Thus we get the diffusion map from the original data to a k-dimensional space which is embedded in the original space.

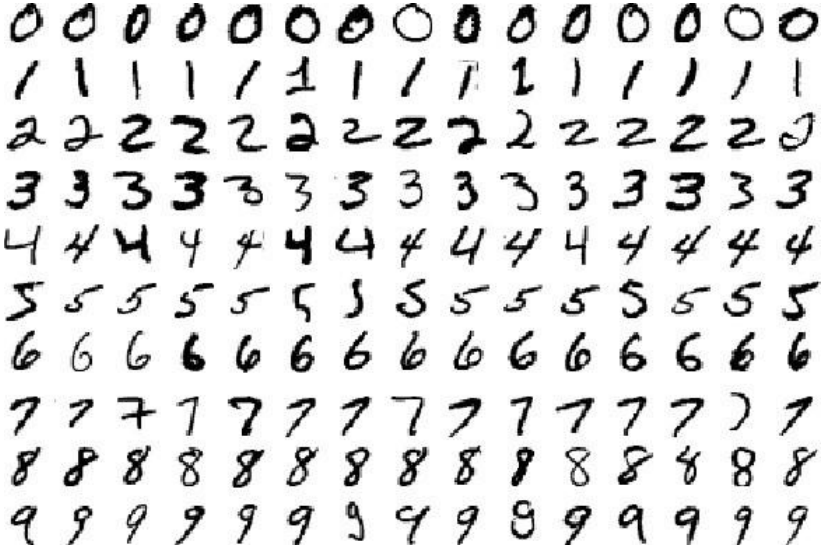
Why it’s better?

Replication Belkin - Niyogi approach

- adjacency graph with n nearest neighbors: W
- graph Laplacian for the adjacency graph: $L = D - W$
- eigendecomposition and define new coordinates: E
- select k the vectors corresponding to the smallest values
- split to unlabeled/labeled
- create classifier

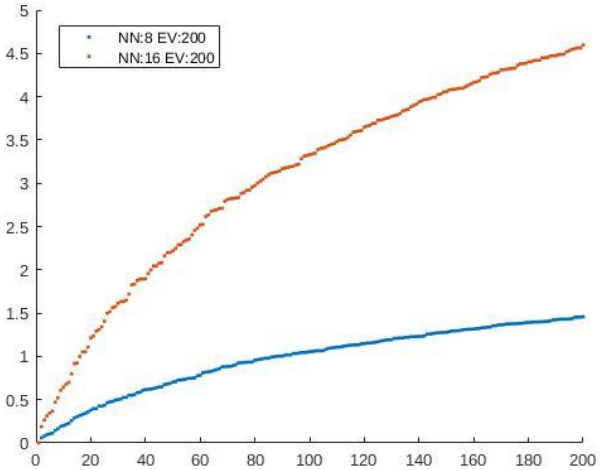
Mnist

- large database of handwritten digits
- contains 60,000 training images and 10,000 testing images
- grayscale images
- 28x28 pixel size converted to flatten array 784 pixel

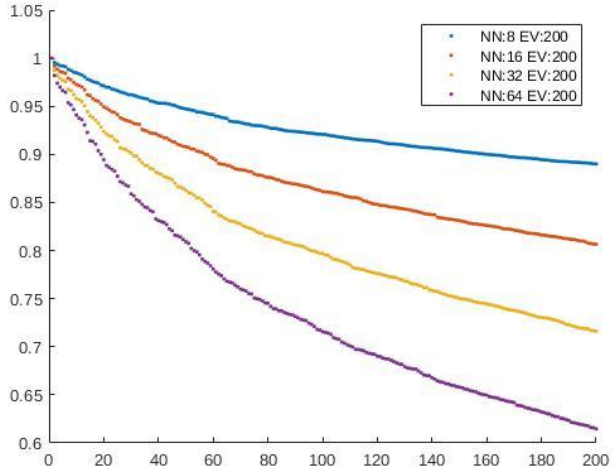


Eigenvalues

Belkin - Niyogi



diffusion maps



Classifier

$\mathbf{E}_{lab}$ - labeled data

$$\mathbf{E}_{lab} = \begin{pmatrix} e_{11} & e_{12} & \dots & e_{1s} \\ e_{21} & e_{22} & \dots & e_{2s} \\ \vdots & \vdots & \ddots & \vdots \\ e_{p1} & e_{p2} & \dots & e_{ps} \end{pmatrix}$$

c - target labels

$$\mathbf{c} = (c_1, \dots, c_s)$$

Solution:

$$\mathbf{a} = \left(\mathbf{E}_{lab}^T \mathbf{E}_{lab}\right)^{-1} \mathbf{E}_{lab}^T \mathbf{c}$$

For the case of several classes, we build a one-against-all classifier for each individual class.

Classifying unlabeled points:

$$c_i = \begin{cases} 1, & \text{if } \sum_{j=1}^{p} e_{ij} a_j \geq 0 \\ -1, & \text{if } \sum_{j=1}^{p} e_{ij} a_j < 0 \end{cases}$$

Accuracy table

Number of eigenvectors

labeled points

	5	10	20	50	100	200
50	0.5130	0.7416	0.8408	0.2920	0.2297	0.1491
100	0.5254	0.7608	0.9036	0.8490	0.2532	0.1928
500	0.5616	0.7612	0.9145	0.9590	0.9569	0.9242
1000	0.5786	0.7554	0.9146	0.9601	0.9624	0.9600
2000	0.5889	0.7506	0.9143	0.9603	0.9631	0.9643
5000	0.5986	0.7485	0.9142	0.9603	0.9633	0.9652

Future directions

- Replace the procedure for computing eigenvectors;
- make a comparative analysis of methods for solving systems of linear equations with sparse matrices;
- parallelize computations;
- apply the developed algorithms to known classification, smoothing tasks.

Applications

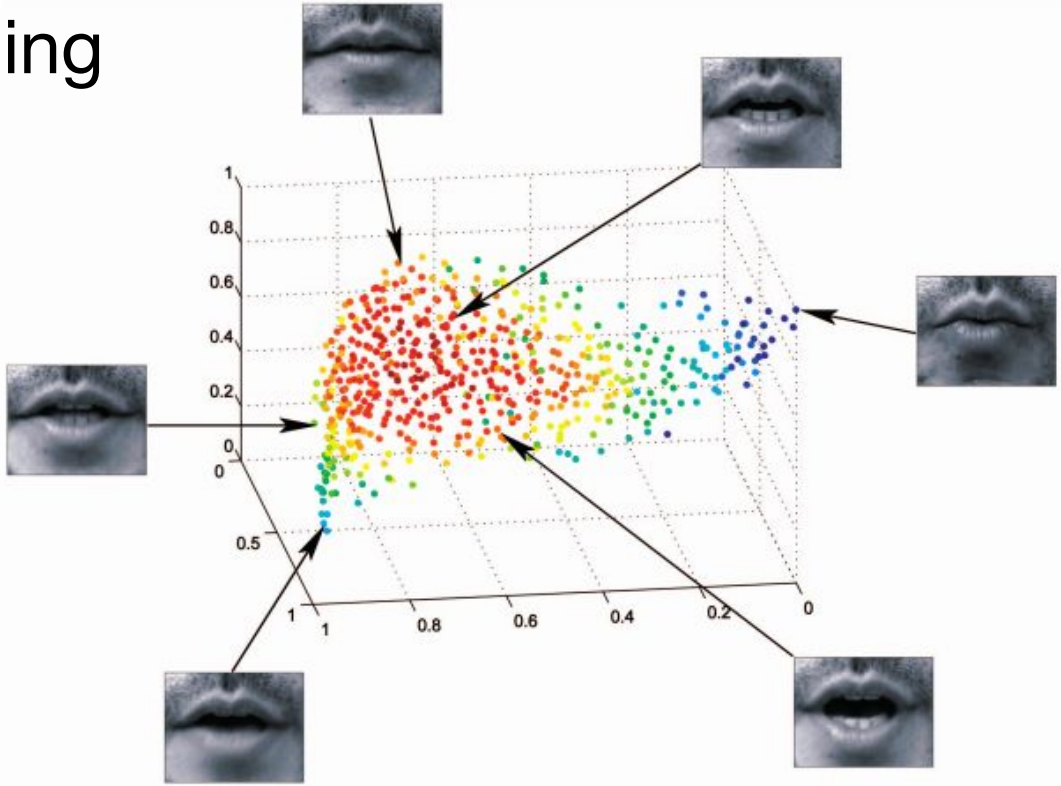
Data Fusion and Multicue Data Matching by Diffusion Maps.

Stéphane Lafon, Yosi Keller, and Ronald R. Coifman (2006)

Application to Lipreading

Task: to perform was isolated-word recognition on a small vocabulary (identification of digits).

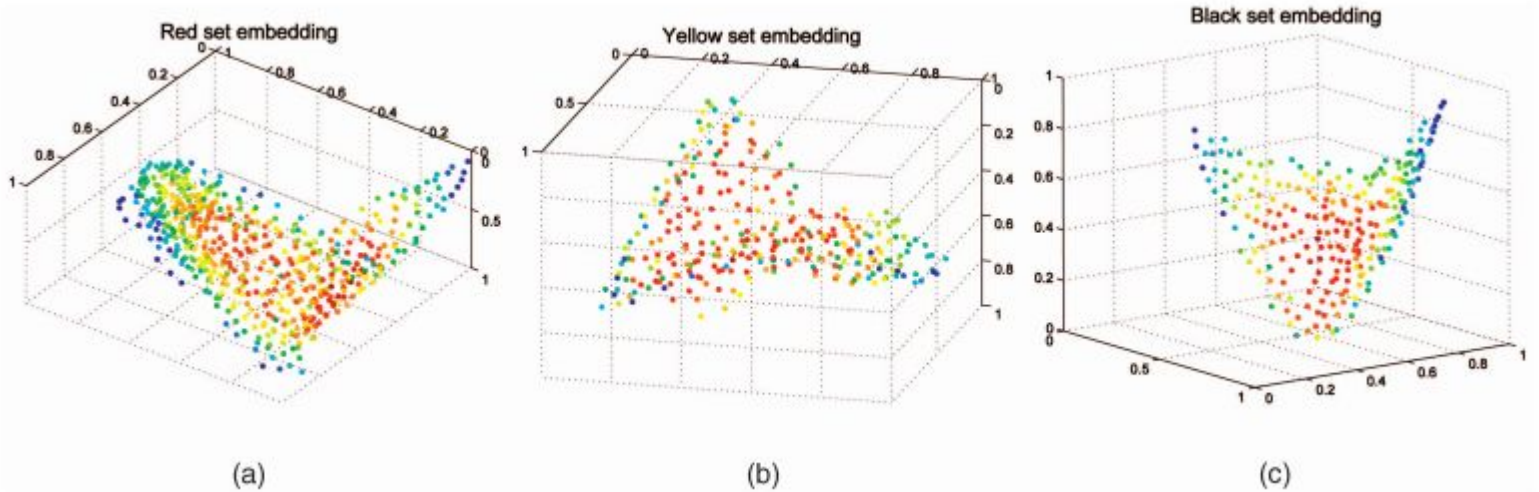
Each word is typically a sequence 25 to 40 frames.



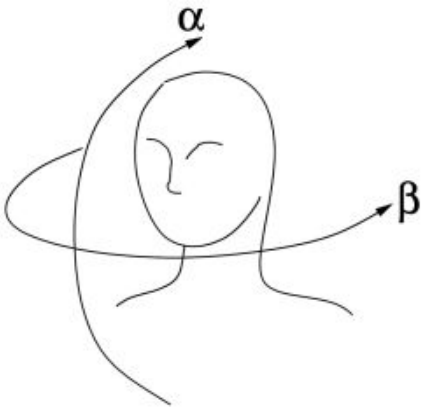
The embedding of the lip data into the top three diffusion coordinates. These coordinates essentially capture two parameters: one controlling the opening of the mouth and the other measuring the portion of teeth that are visible.

Synchronization of Head Movement Data

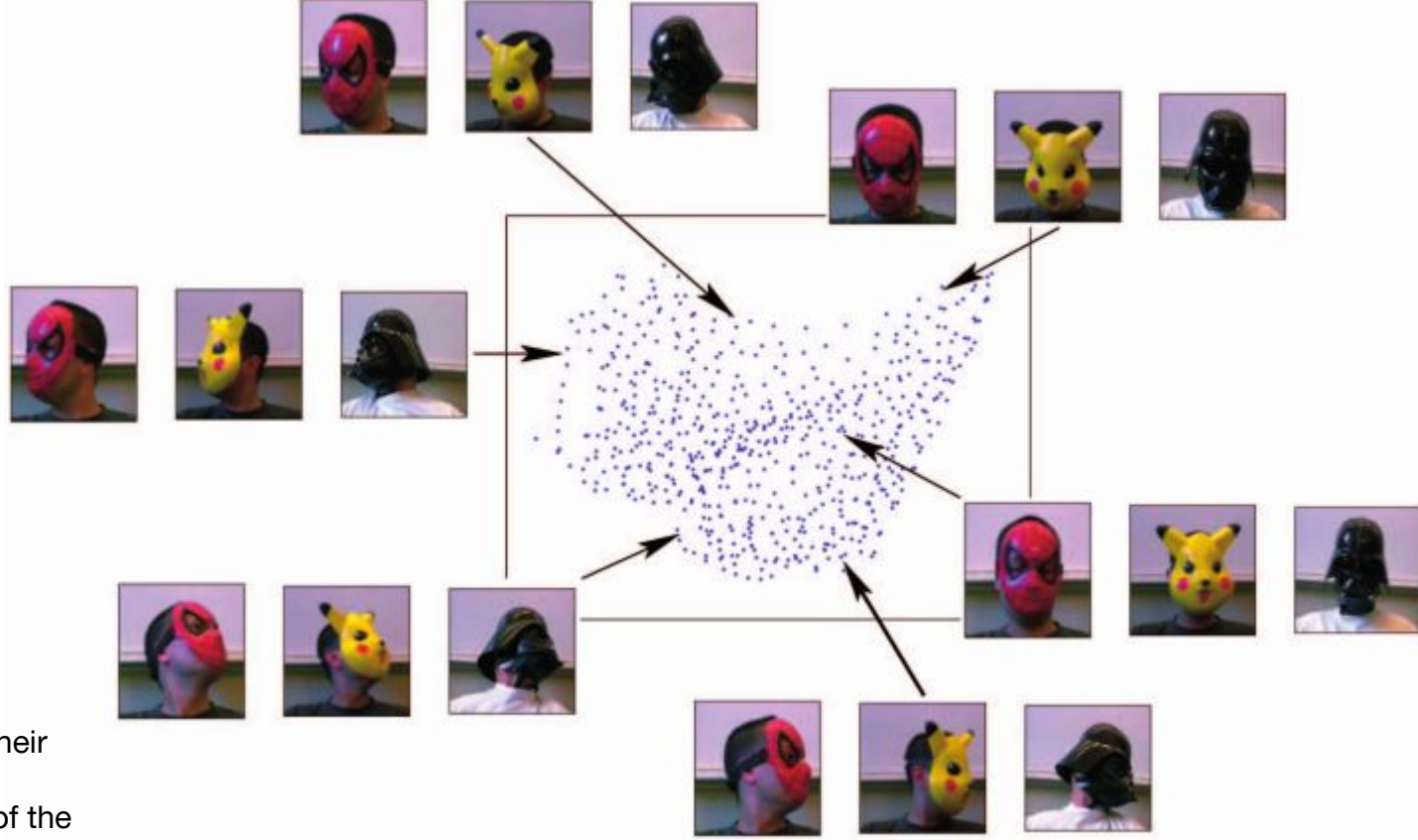
Three movies of subjects wearing successively a yellow, red, and black mask were recorded. Each subject was asked to move their head in front of the camcorder. Then the three sets consisting of all frames of each movie were considered. The goal was to synchronize the movements of the different masks by aligning the three diffusion embeddings



The embedding of each set in the first three diffusion coordinates. The color encodes the density of points. All three sets share this butterfly-shaped embedding.



Each subject essentially moved their head along the two angles
and. There was almost no tilting of the head.
Hence, the data points approximately lie on a submanifold of dimension 2.



The embedding of the YELLOW set in three diffusion coordinates and the various corresponding images after alignment of the RED and
BLACK graphs to YELLOW.

Spatial-spectral operator theoretic methods for hyperspectral image classification

John J. Benedetto, Wojciech Czaja, Julia Dobrosotskaya,
Timothy Doster, Kevin Duke (2016)

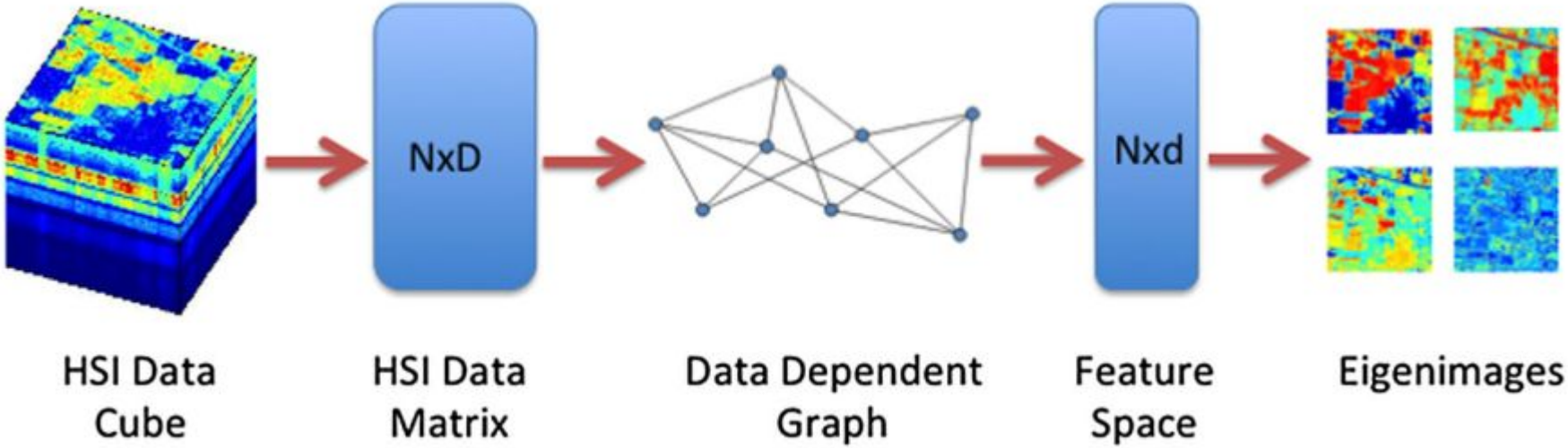
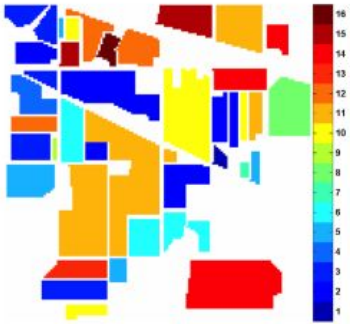


Diagram depicting the Laplacian Eigenmaps algorithm stages.

Given a m × n × D hyperspectral data cube we reshape this to form a N × D data matrix.
A data dependent graph is then learned using kNN.
A cost function is minimized to produce a lower dimensional feature space representation of the data in the form of a N × d matrix.
Each column of this matrix can be reshaped to form an eigenimage

Dataset

(a)



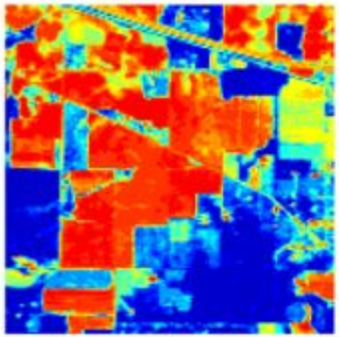
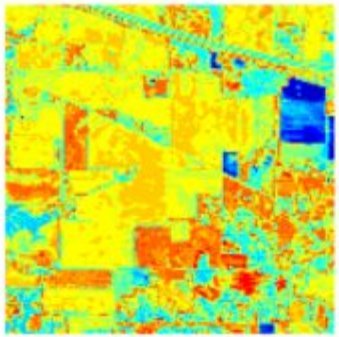
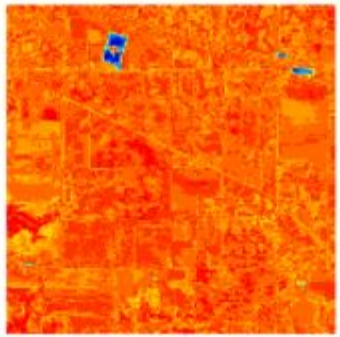
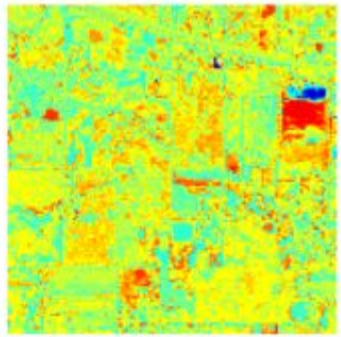
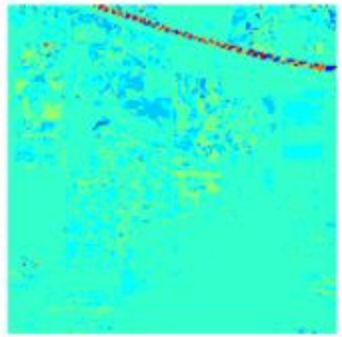
(b)

#	Class	Samples
1	Alfalfa	46
2	Corn-notill	1428
3	Corn-mintill	830
4	Corn	237
5	Grass-pasture	483
6	Grass-trees	730
7	Grass-pasture-mowed	28
8	Hay-windrowed	478
9	Oats	20
10	Soybean-notill	972
11	Soybean-mintill	2455
12	Soybean-clean	593
13	Wheat	205
14	Woods	1265
15	Buildings-Grass-Trees-Drives	386
16	Stone-Steel-Towers	93

(c)

a The color composite image of Indian Pines.
b Ground truth for Indian Pines.
c Ground truth descriptions and sample count for Indian Pines

Eigenmaps that correspond to the eigenvectors produced by the Laplacian Eigenmaps algorithm were reshaped to form a matrix corresponding to the original dimensions of the hyper-spectral Image and rescaled linearly to produce a desired colormap. These eigenmaps can be interpreted as highlighting similar features one- by-one in the various eigenimages

<- classification results for Spectral Laplacian Eigenmaps.

Class map using Spectral Laplacian Eigenmaps

Ground-truth class map

Image Completion by Diffusion Maps and Spectral Relaxation

Shai Gepshtein and Yosi Keller

Image inpainting algorithms aim to fill the missing data of an image or a video sequence in a visually plausible way, such that the insertion is not easily detectable by a common unsuspecting viewer

The image completion problem lies at the intersection of computer graphics, image, and signal processing. Formally, given a corrupted image I and a “hole” region mask H marking the unknown area, the goal of image inpainting is to fill in H to form a visually plausible image I.

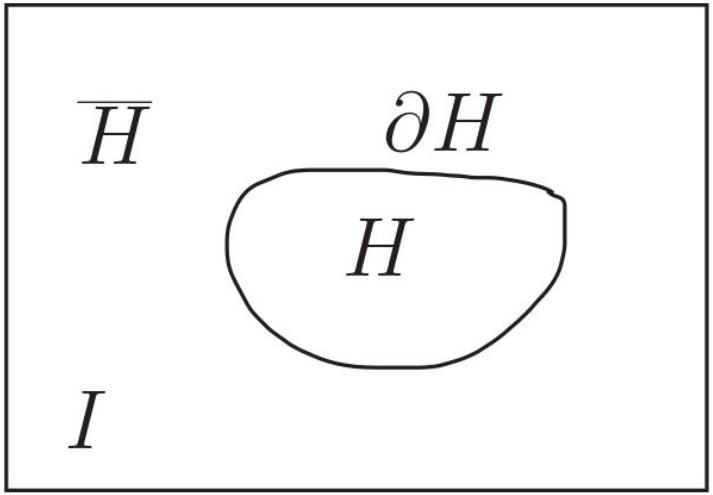


Image completion problem. We are given an image I , the unknown part H , and its complementary ~~H~~ , such that I = H ∪ ~~H~~
∂H is the boundary of H.

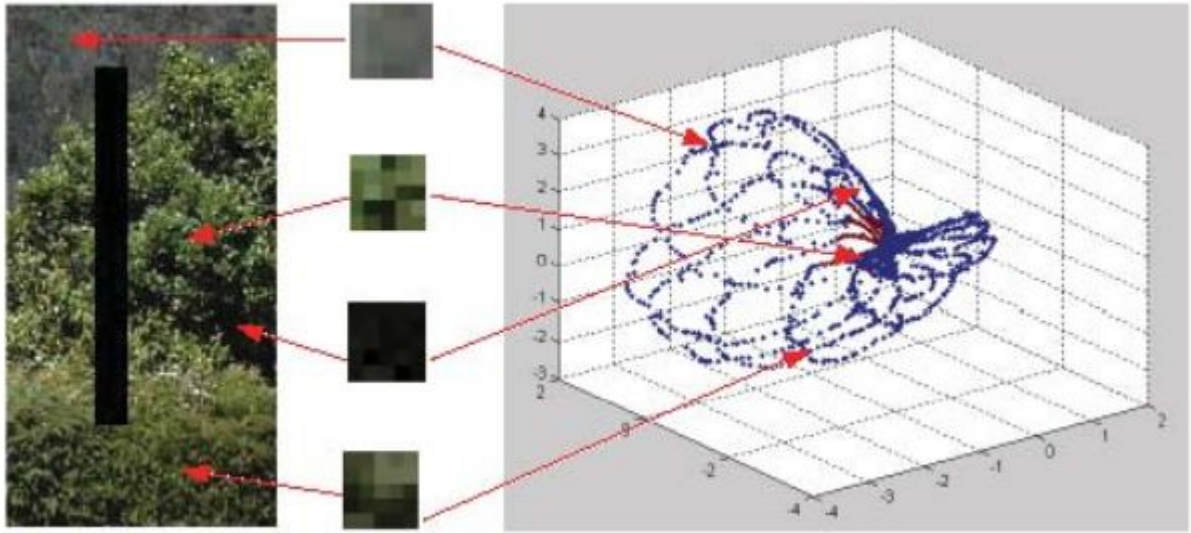
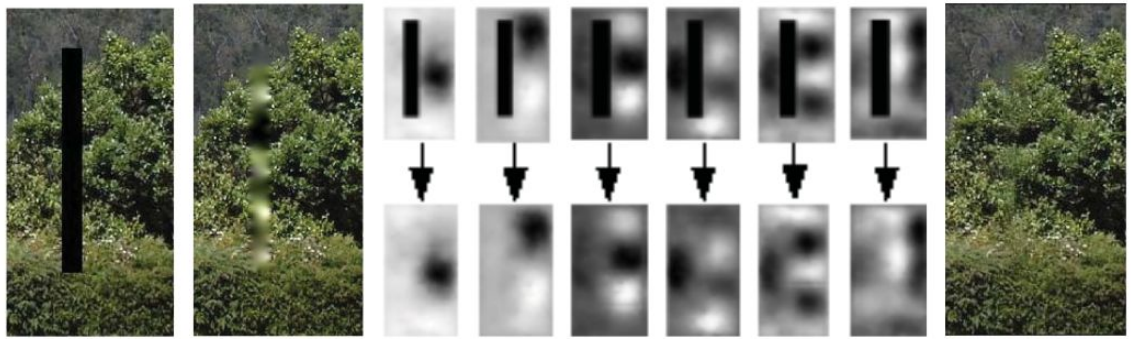
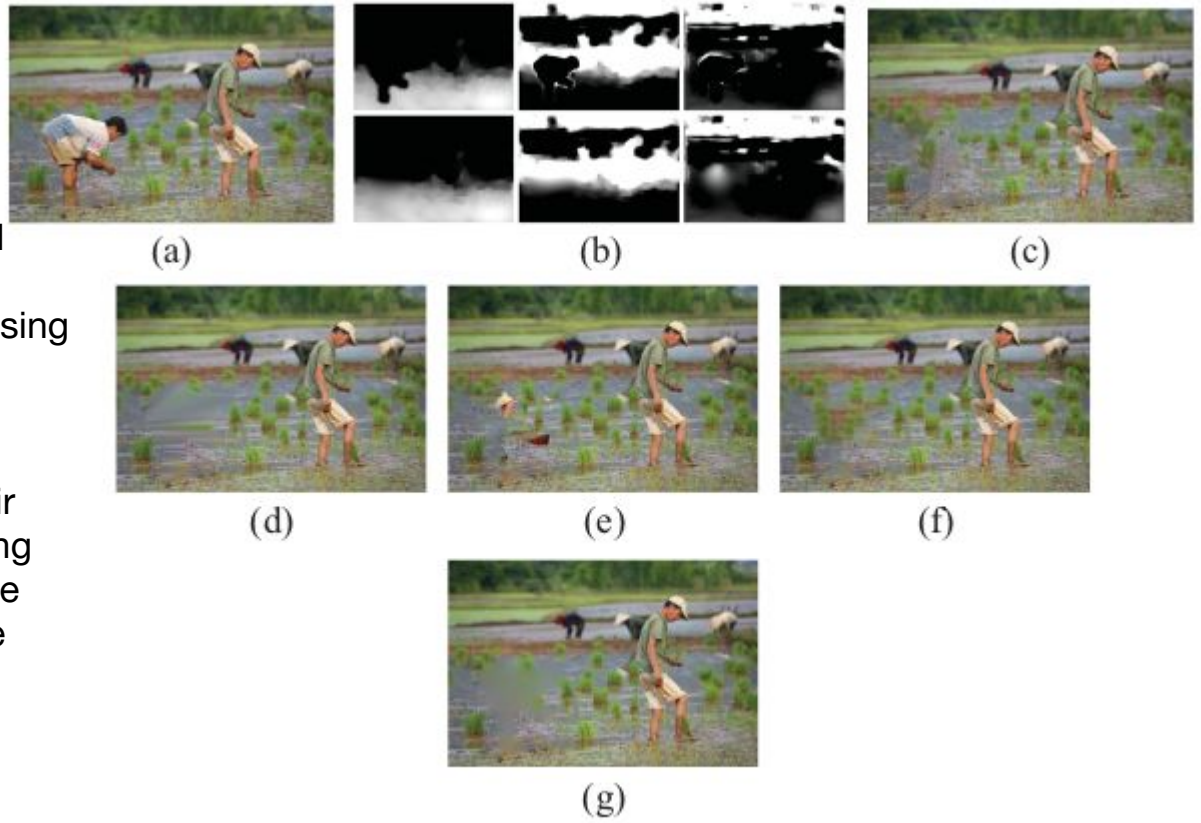


Image and its embedding. Patches with similar texture are mapped to close locations on the manifold, using an embedding based on LBP texture descriptors

Image inpainting



(a) Source image to be inpainted.
(b) Source image inpainted by an isotropic heat equation.
(c) Leading eigenvectors inpainted using an isotropic heat equation.
(d) Image inpainted through computing the inverse diffusion mapping of (c)



Inpainting results of the Rice field image.
The embedding was computed using the LBP texture descriptor.

(a) Original image.
(b) Leading eigenvectors and their inpaintings. Upper row: the leading embedding eigen-vectors with the unknown part H. Bottom row: the inpainted eigenvectors.
(c) Proposed scheme.
(d) Tschumperle et al.
(e) Best-exemplar
(f) Komodakis et al.
(g) Isotropic heat equation.

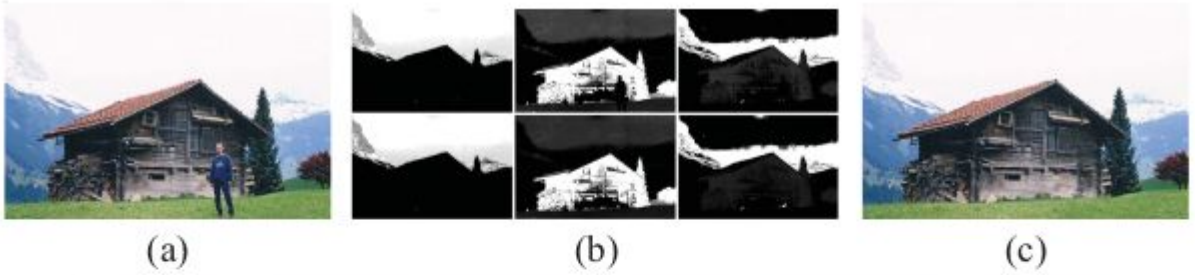
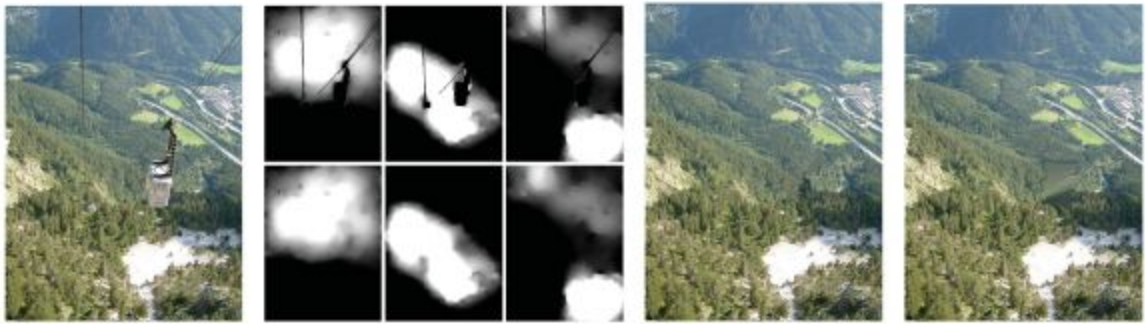


Image completion through the 5-D texture descriptor of and best-exemplar inpainting of the eigenvectors.
(a) Original image.
(b) Leading embedding eigenvectors and their inpaintings.
(c) Proposed scheme.
(d) Tschumperle et al.
(e) Best exemplar
(f) Komodakis et al.
(g) Isotropic heat equation.



(a) (b) (c) (d)

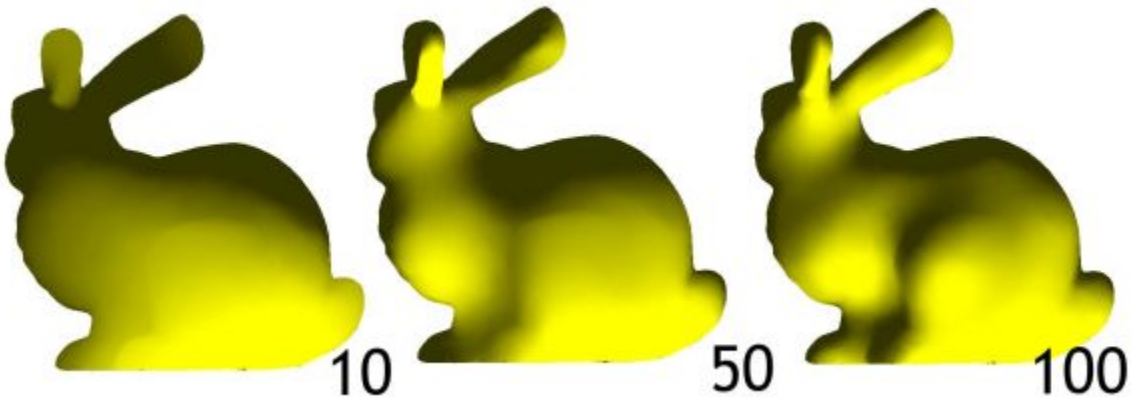
(e) (f) (g)

Laplace-Beltrami Eigenfunctions Towards an algorithm that “understands” geometry

Bruno Lévy

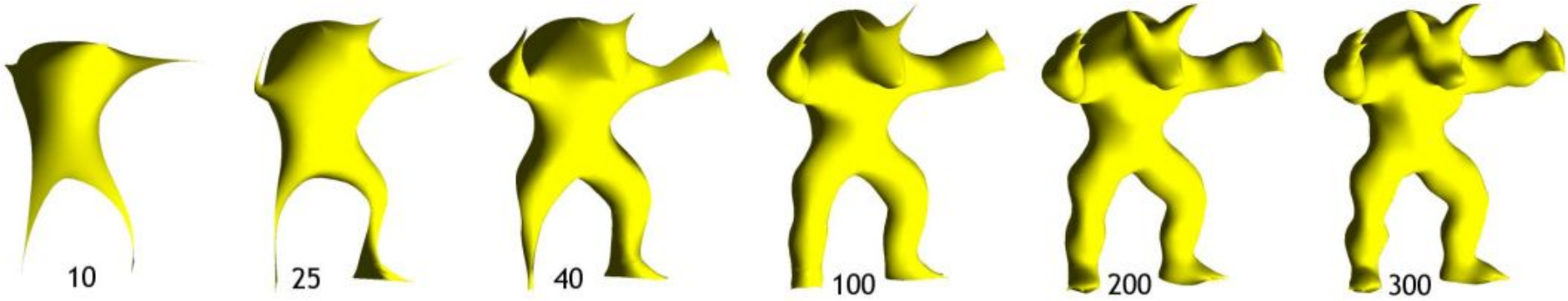
Signal processing on surfaces

Once the eigenfunction basis is computed, it can be used to represent various functions on the surface.



Reconstructing a signal (here the normal vector) using the eigenfunction basis.
The variations of the normal vector over the bunny are well approximated by 100 coefficients.

Geometry processing



Geometric reconstructions obtained with increasing number of eigenfunctions

Registration and pose transfer

1. Compute the eigenfunction bases (Φ_i) of the Armadillo and (Φ'_i) of Homer;

2. Project the geometry onto each eigenfunction Φ_i and Φ'_i

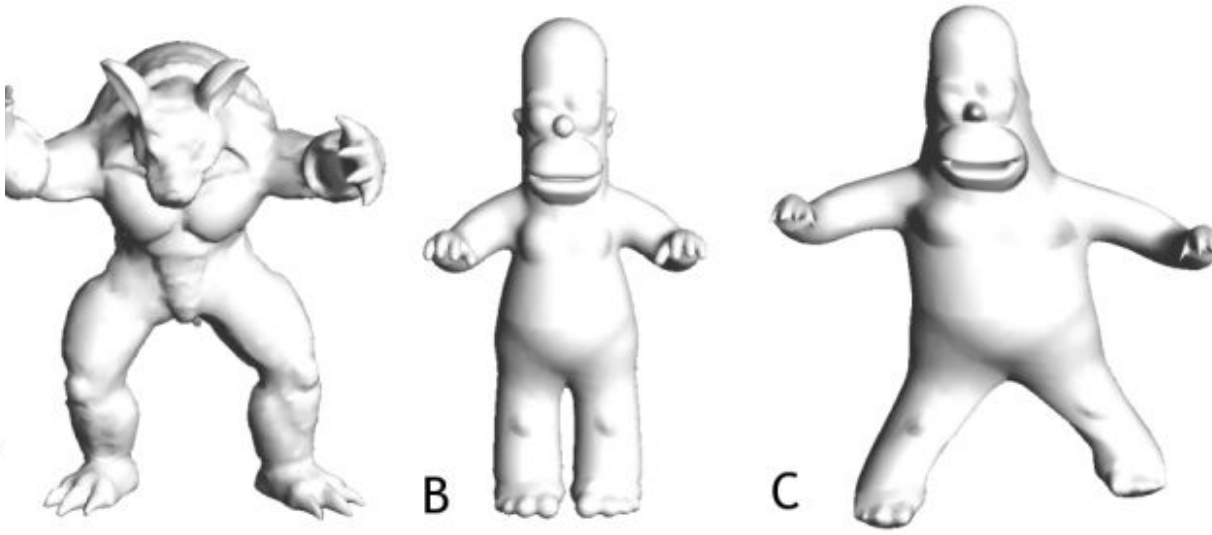
$$\begin{aligned} \alpha_i^x &\leftarrow \textstyle\sum_j x_j \Phi_i(j) \\ \alpha_i^y &\leftarrow \textstyle\sum_j y_j \Phi_i(j) \\ \alpha_i^z &\leftarrow \textstyle\sum_j z_j \Phi_i(j) \end{aligned}$$

$$\begin{aligned} \beta_i^x &\leftarrow \textstyle\sum_j x'_j \Phi'_i(j) \\ \beta_i^y &\leftarrow \textstyle\sum_j y'_j \Phi'_i(j) \\ \beta_i^z &\leftarrow \textstyle\sum_j z'_j \Phi'_i(j) \end{aligned}$$

where $p_j = (x_j, y_j, z_j)$ denote the vertices of the Armadillo and $q_j = (x'_j, y'_j, z'_j)$ denote the vertices of Homer;

3. Reconstruct the signal for each vertex j, using the α's for the low frequencies and the β's for the high frequencies:

$$\begin{aligned} x_j &\leftarrow \textstyle\sum_{i=1}^{5} \alpha_i^x \Phi'_i(j) + \sum_{i=6}^{n} \beta_i^x \Phi'_i(j) \\ y_j &\leftarrow \textstyle\sum_{i=1}^{5} \alpha_i^y \Phi'_i(j) + \sum_{i=6}^{n} \beta_i^y \Phi'_i(j) \\ z_j &\leftarrow \textstyle\sum_{i=1}^{5} \alpha_i^z \Phi'_i(j) + \sum_{i=6}^{n} \beta_i^z \Phi'_i(j) \end{aligned}$$



Pose transfer: the 5 first coefficients of the Armadillo (A) were copied to Homer (B) and adapted its pose to the Armadillo (C)