Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models

Quan Wang
ECSE, Rensselaer Polytechnic Institute
May 13, 2011

Main Work

- Studied the theories of kernel PCA and Active Shape Models (ASMs)
- Implemented kernel PCA and used it to design nonlinear ASMs
- Tested kernel PCA for
  - Synthetic data classification
  - Human face images classification
  - Building nonlinear ASMs for human face models
- Designed a parameter selection method

Contents

- PCA and ASMs
- Kernel PCA
  - Constructing the Kernel Matrix
  - Reconstructing Pre-Images
  - Pre-Images for Gaussian Kernels
- Experiments
  - Pattern Classification for Synthetic Data
  - Classification for Aligned Human Face Images
  - Kernel PCA Based Active Shape Models
- Discussion
  - Parameter Selection

PCA Review

- Find a linear projection $\mathbf{y} = \mathbf{A}\mathbf{x}$ such that the projected data has the largest variance: $\max_{\mathbf{A}} \mathrm{tr}(\mathbf{S}_{\mathbf{y}})$

- Solution: $\mathbf{S}_{\mathbf{x}}\mathbf{u}_i = \lambda_i \mathbf{u}_i$ where $\mathbf{A} = [\mathbf{u}_1^t, \ldots, \mathbf{u}_M^t]^t$

- Reconstruction of original data:

$$\mathbf{x} = \sum_{i=1}^{M} a_i \mathbf{u}_i \quad \text{where} \quad a_i = (\mathbf{x} \cdot \mathbf{u}_i)$$

Active Shape Models (ASMs)

- ASM is a statistical model of the shape of objects
- It iteratively deforms to fit the object in new images
- Shapes are constrained by Point Distribution Models (PDM)

Point Distribution Model (PDM)

- A shape is described by *n* landmark points, thus we have our shape vector:

$$\mathbf{x} = (x_1, y_1, \dots, x_n, y_n)^T$$

- With *s* training data, we have the mean shape and covariance:

$$\bar{\mathbf{x}} = \frac{1}{s}\sum_{i=1}^{s}\mathbf{x}_i \qquad \mathbf{S} = \frac{1}{s-1}\sum_{i=1}^{s}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T$$

Principal Component Analysis (PCA)

- The eigenvectors corresponding to the t largest eigenvalues λ_i of $\mathbf{S}$ are retained in a matrix $\mathbf{P} = (\mathbf{p}_1, \dots, \mathbf{p}_t)$
- A shape can now be approximated by:

  $\mathbf{x} \approx \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}$

- $\mathbf{b}$ is a vector of elements containing the model parameters

$$\mathbf{S} \xrightarrow{\text{PCA}} \mathbf{P}$$

Model Parameters

- When fitting the model to a set of points, the values of $\mathbf{b}$ are constrained to lie within the range $\pm m\sqrt{\lambda_i}$, where usually m has a value between two and three.

Demos

- PDM of a resistor

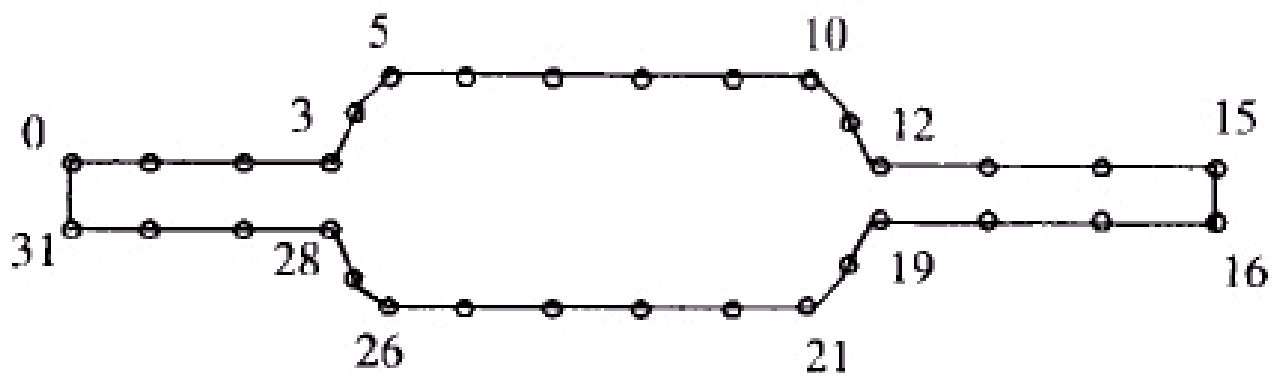


Figure 1. Point index of a resistor.

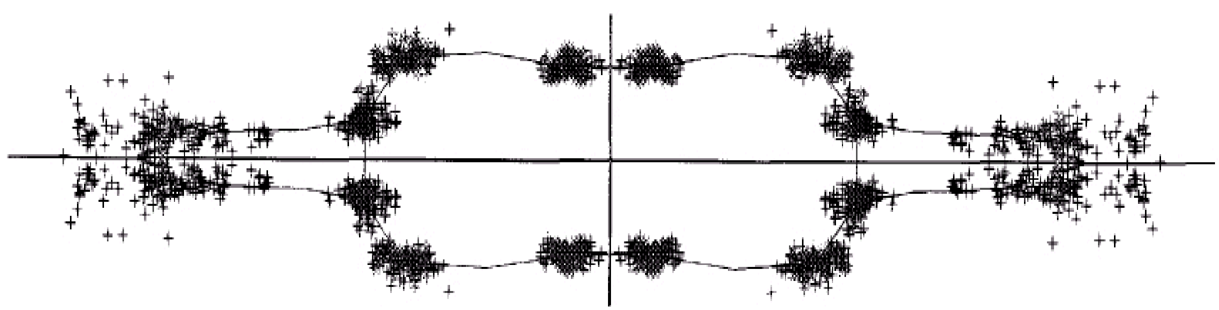
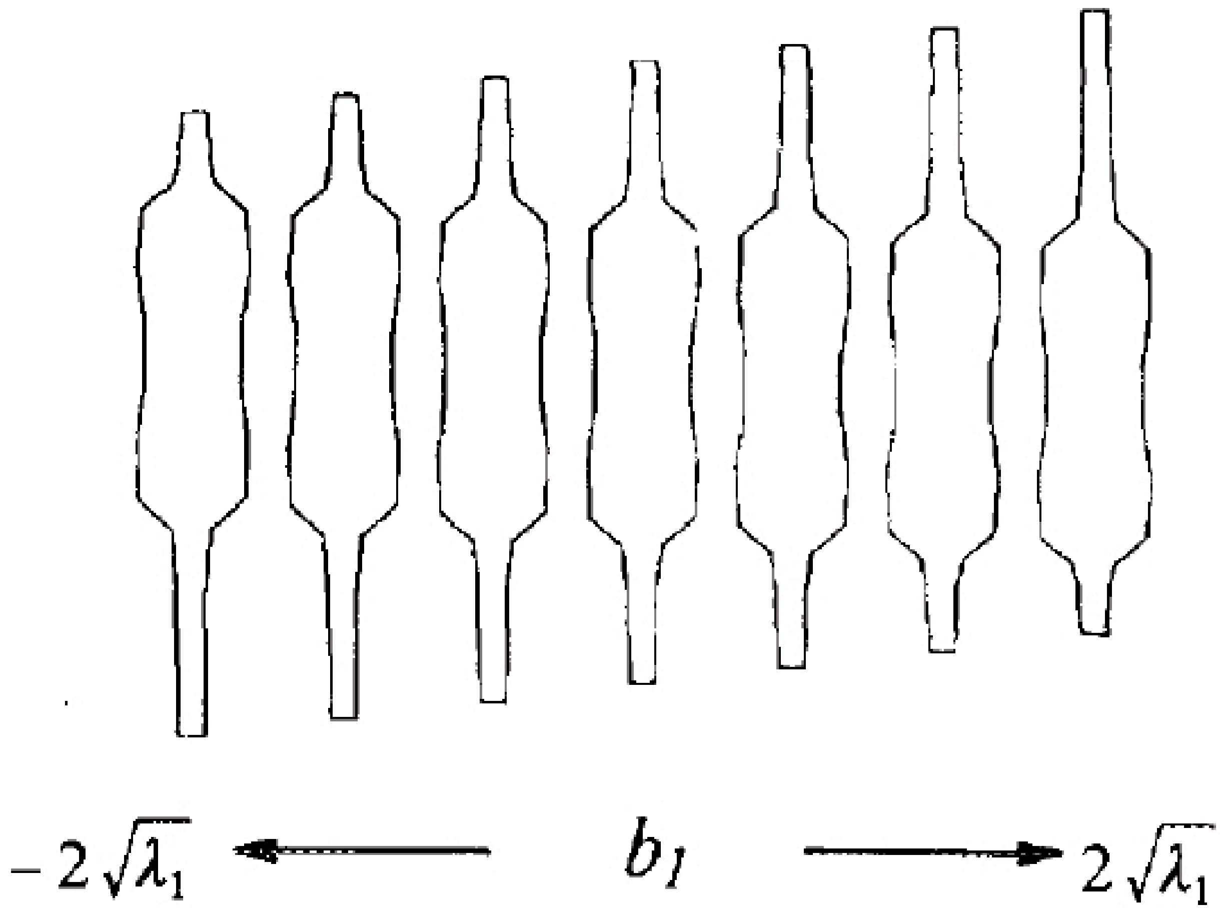
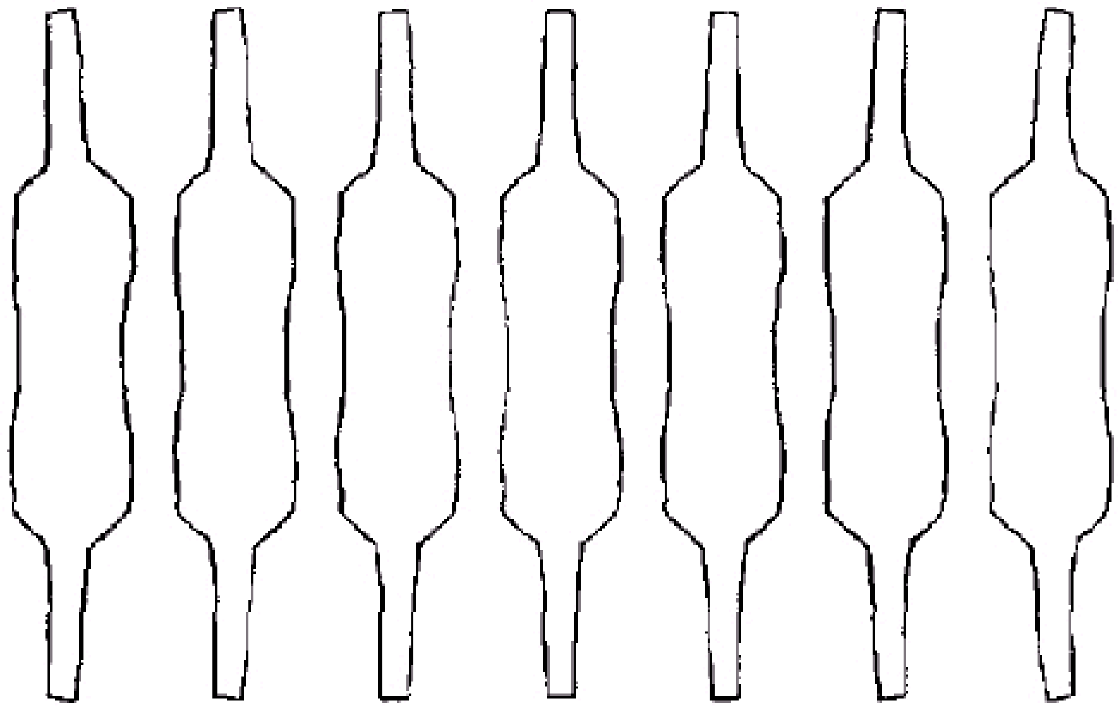


Figure 2. PDM of a resistor.

Demos



Demos



$-2\sqrt{\lambda_3}$ ⟵ b_3 ⟶ $2\sqrt{\lambda_3}$

Kernel PCA

- Traditional PCA: only for linear projection
- Kernel PCA: nonlinear dimensionality reduction
- Basic idea:
  - Find a nonlinear projection $\phi(\mathbf{x})$ from x to an *M* dimensional feature space
  - Apply eigenvector analysis for

$$\mathbf{C} = \frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^{\mathrm{T}}$$

assuming $\sum_n \phi(\mathbf{x}_n) = \mathbf{0}$

Constructing Kernel Matrix

$$\mathbf{C}\mathbf{v}_i = \lambda_i \mathbf{v}_i \qquad \mathbf{C} = \frac{1}{N}\sum_{n=1}^{N} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^{\mathrm{T}}$$

$$\Downarrow$$

$$\frac{1}{N}\sum_{n=1}^{N} \phi(\mathbf{x}_n)\{\phi(\mathbf{x}_n)^{\mathrm{T}}\mathbf{v}_i\} = \lambda_i \mathbf{v}_i$$

$$\Downarrow$$

$$\mathbf{v}_i = \sum_{n=1}^{N} a_{in}\phi(\mathbf{x}_n)$$

$$\Downarrow$$

$$\frac{1}{N}\sum_{n=1}^{N} \phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^{\mathrm{T}} \sum_{m=1}^{N} a_{im}\phi(\mathbf{x}_m) = \lambda_i \sum_{n=1}^{N} a_{in}\phi(\mathbf{x}_n)$$

Constructing Kernel Matrix

$$\frac{1}{N}\sum_{n=1}^{N}\phi(\mathbf{x}_n)\phi(\mathbf{x}_n)^{\mathrm{T}}\sum_{m=1}^{N}a_{im}\phi(\mathbf{x}_m)=\lambda_i\sum_{n=1}^{N}a_{in}\phi(\mathbf{x}_n)$$

Multiply both sides by $\phi(\mathbf{x}_l)^{\mathrm{T}}$

and define the kernel $k(\mathbf{x}_n,\mathbf{x}_m)=\phi(\mathbf{x}_n)^{\mathrm{T}}\phi(\mathbf{x}_m)$

$$\frac{1}{N}\sum_{n=1}^{N}k(\mathbf{x}_l,\mathbf{x}_n)\sum_{m=1}^{N}a_{im}k(\mathbf{x}_n,\mathbf{x}_m)=\lambda_i\sum_{n=1}^{N}a_{in}k(\mathbf{x}_l,\mathbf{x}_n)$$

$$\Downarrow$$

$$\mathbf{K}^2\mathbf{a}_i=\lambda_i N\mathbf{K}\mathbf{a}_i \Rightarrow \mathbf{K}\mathbf{a}_i=\lambda_i N\mathbf{a}_i$$

Constructing Kernel Matrix

- Extracting kernel PCA features:

$$y_i(\mathbf{x}) = \phi(\mathbf{x})^{\mathrm{T}}\mathbf{v}_i = \sum_{n=1}^{N} a_{in} k(\mathbf{x}, \mathbf{x}_n)$$

- If $\phi(\mathbf{x}_n)$ does not have zero mean:

$$\widetilde{\mathbf{K}} = \mathbf{K} - \mathbf{1}_N\mathbf{K} - \mathbf{K}\mathbf{1}_N + \mathbf{1}_N\mathbf{K}\mathbf{1}_N$$

where $\mathbf{1}_N = \frac{1}{N}\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}$

Commonly used kernels

- Polynomial kernel:

$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^{\mathrm{T}}\mathbf{y})^d$$

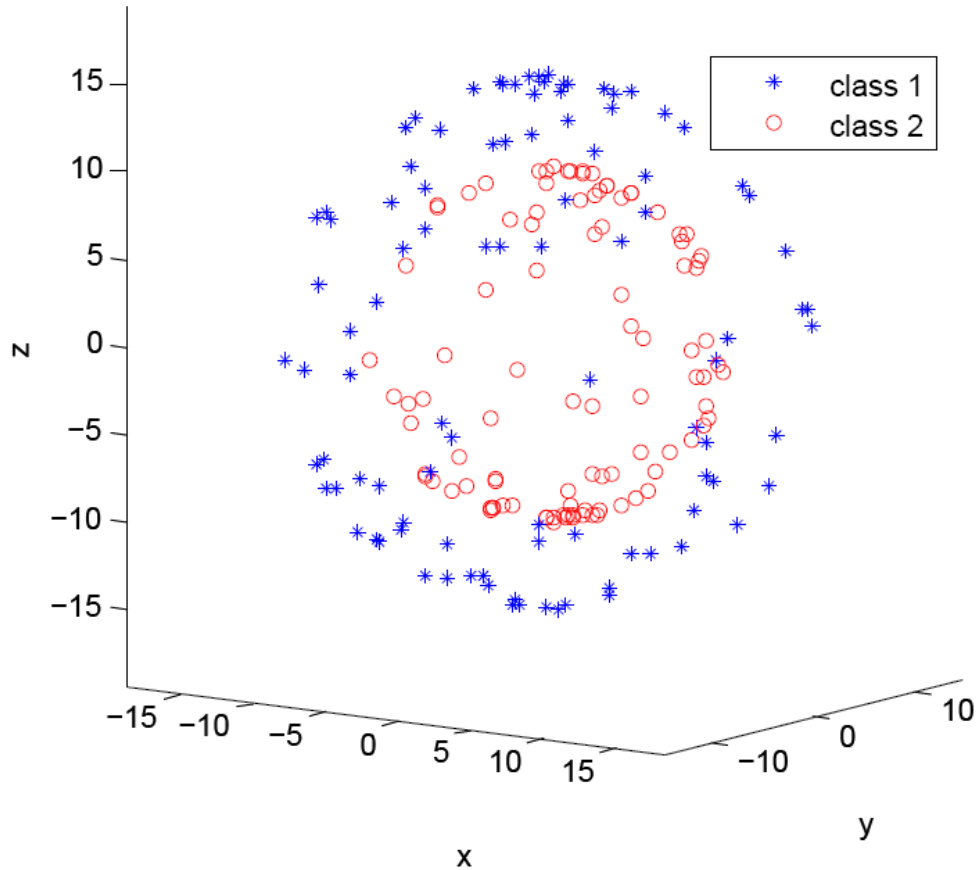
$$k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^{\mathrm{T}}\mathbf{y} + c)^d$$

- Gaussian kernel:

$$k(\mathbf{x}, \mathbf{y}) = \exp\left(-\|\mathbf{x} - \mathbf{y}\|^2 / 2\sigma^2\right)$$

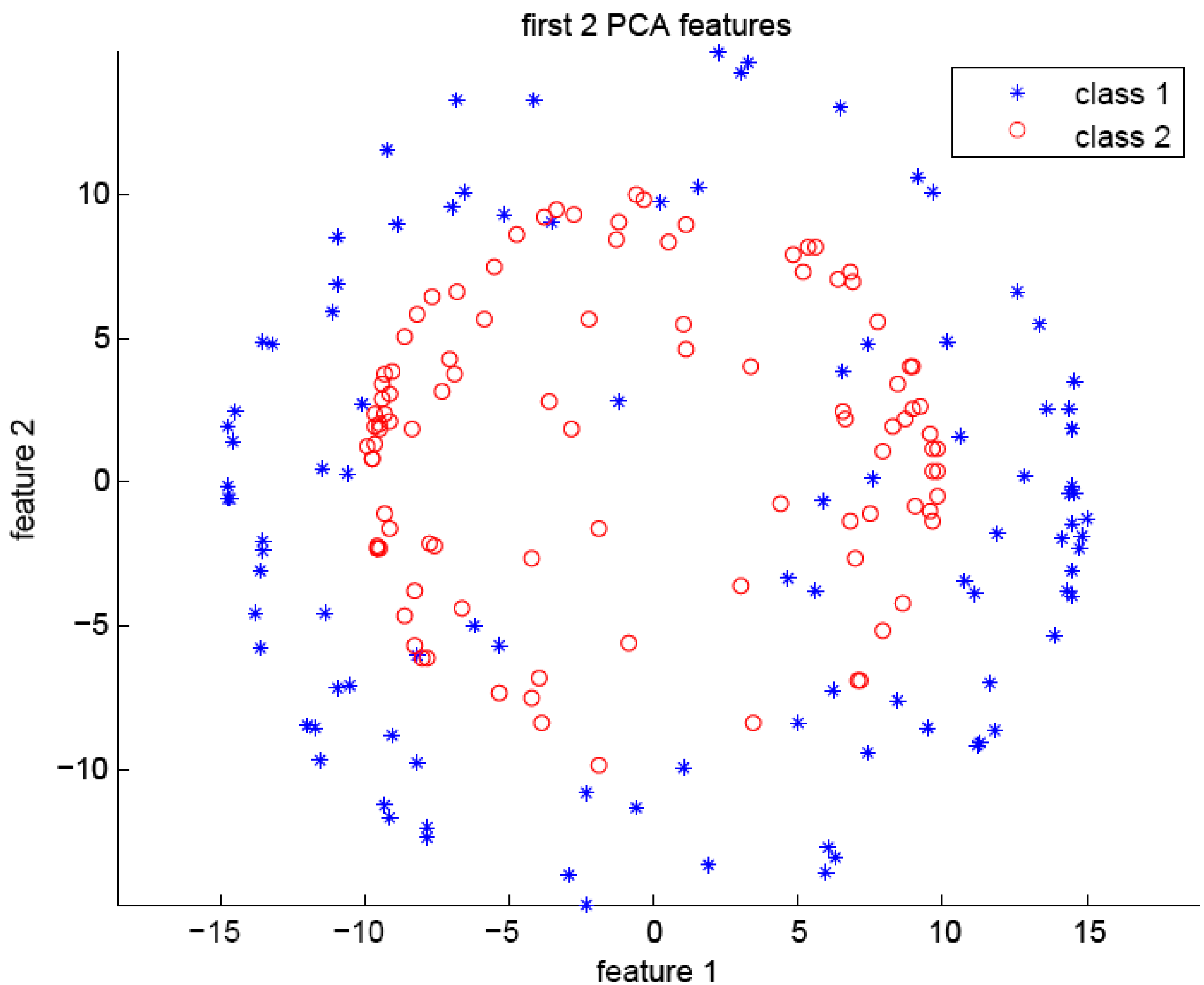
Experiment on Synthetic Data

- Data: data points uniformly distributed on two-concentric-spheres



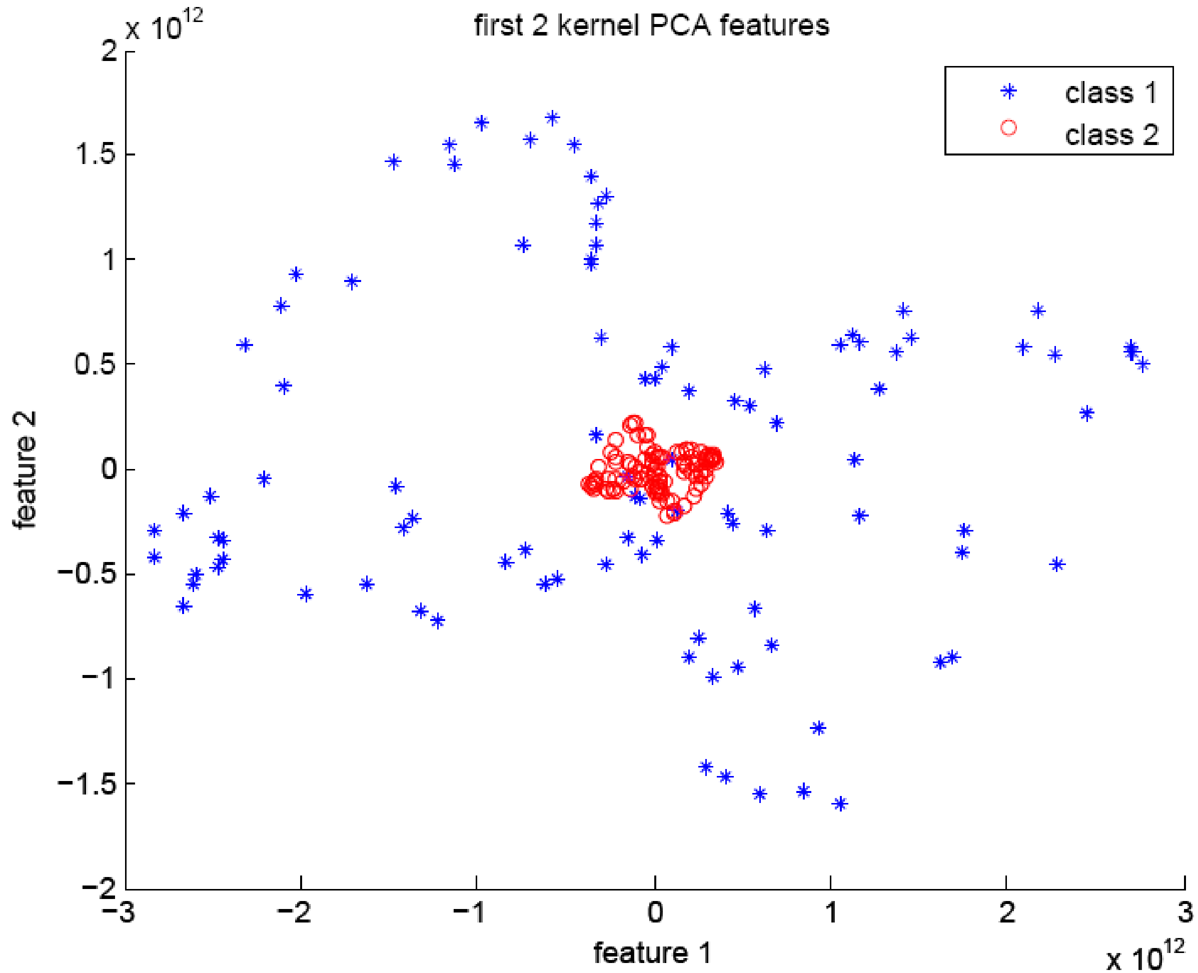
Experiment on Synthetic Data

- First two PCA features



Experiment on Synthetic Data

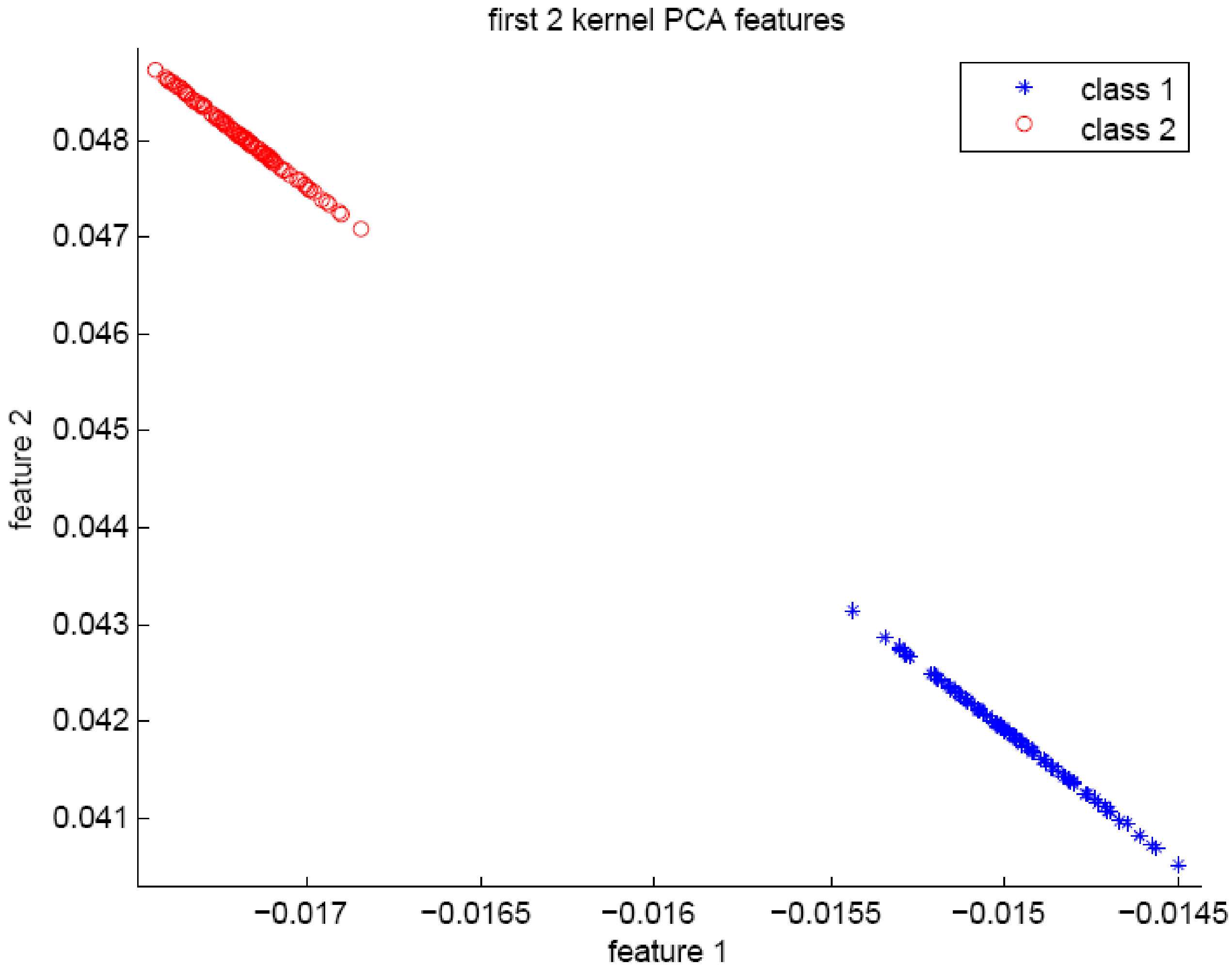
- First two polynomial kernel PCA features



d=5

Experiment on Synthetic Data

- First two Gaussian kernel PCA features



$\sigma = 20$

Experiment on Human Face Images

- Use PCA / kernel PCA to extract the 10 most significant features
- Use simplest linear classifier to classify the face images of two subjects
- Data: Yale Face Database B

Experiment on Human Face Images

- We use Gaussian kernel PCA with $\sigma = 45675$ (will talk about parameter selection later)

Error Rate	PCA	Kernel PCA
Training Data Test Data	0.0686 0.1923	0.0588 0.1154

Reconstructing Pre-Images for Kernel PCA

- Reconstruction of pre-images for PCA is easy:

$$\mathbf{x} = \sum_{i=1}^{M} a_i \mathbf{u}_i \qquad\qquad a_i = (\mathbf{x} \cdot \mathbf{u}_i)$$

- This is very difficult for kernel PCA, but for Gaussian kernel PCA, we have specific algorithms to find an approximation of the pre-image $\mathbf{z}$

Pre-Images for Gaussian Kernel PCA

$$\mathbf{z} = \frac{\sum_{i=1}^{N} \gamma_i \exp\left(-\|\mathbf{z} - \mathbf{x}_i\|^2 / 2\sigma^2\right) \mathbf{x}_i}{\sum_{i=1}^{N} \gamma_i \exp\left(-\|\mathbf{z} - \mathbf{x}_i\|^2 / 2\sigma^2\right)}$$

$$\gamma_i = \sum_{k=1}^{n} y_k a_{ik}$$

Pre-Images for Gaussian Kernel PCA

- Iterative algorithm:

$$\mathbf{z}_{t+1} = \frac{\sum_{i=1}^{N} \gamma_i \exp\left(-\|\mathbf{z}_t - \mathbf{x}_i\|^2 / 2\sigma^2\right) \mathbf{x}_i}{\sum_{i=1}^{N} \gamma_i \exp\left(-\|\mathbf{z}_t - \mathbf{x}_i\|^2 / 2\sigma^2\right)}$$

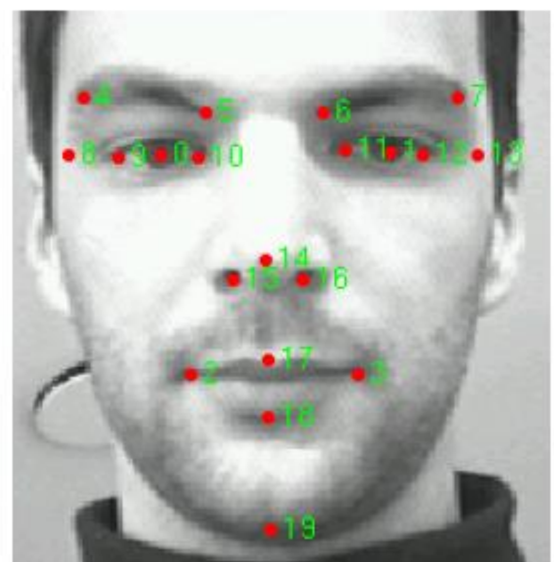
Kernel PCA based ASMs

- ASMs use traditional PCA to describe the deformation patterns of the shape of an object

- Once we are able to extract kernel PCA features and reconstruct pre-images of kernel PCA, we can design kernel PCA based ASMs

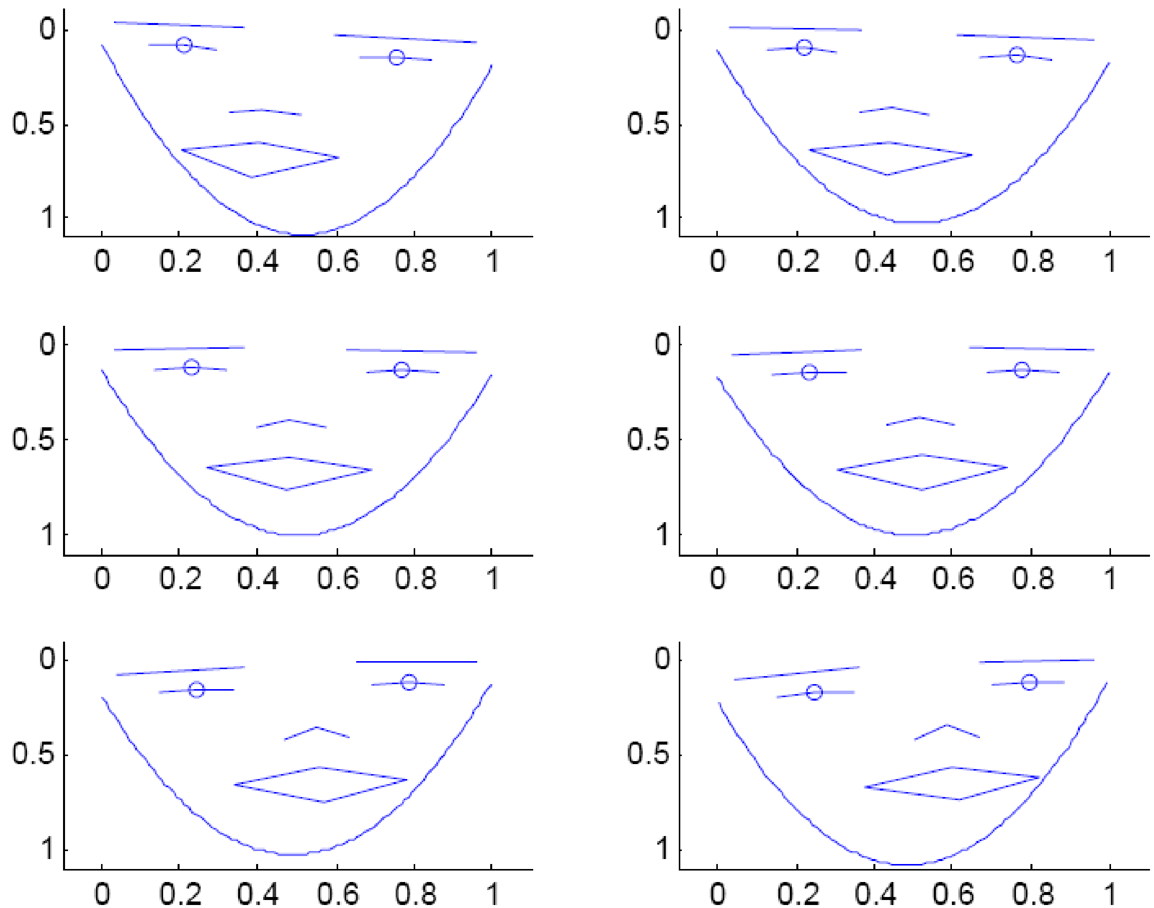
ASMs for Human Face Images

- We use Tim Cootes' manually annotated points of 1521 human face images from the BioID database
- Then we apply ASMs and Gaussian kernel PCA based ASMs



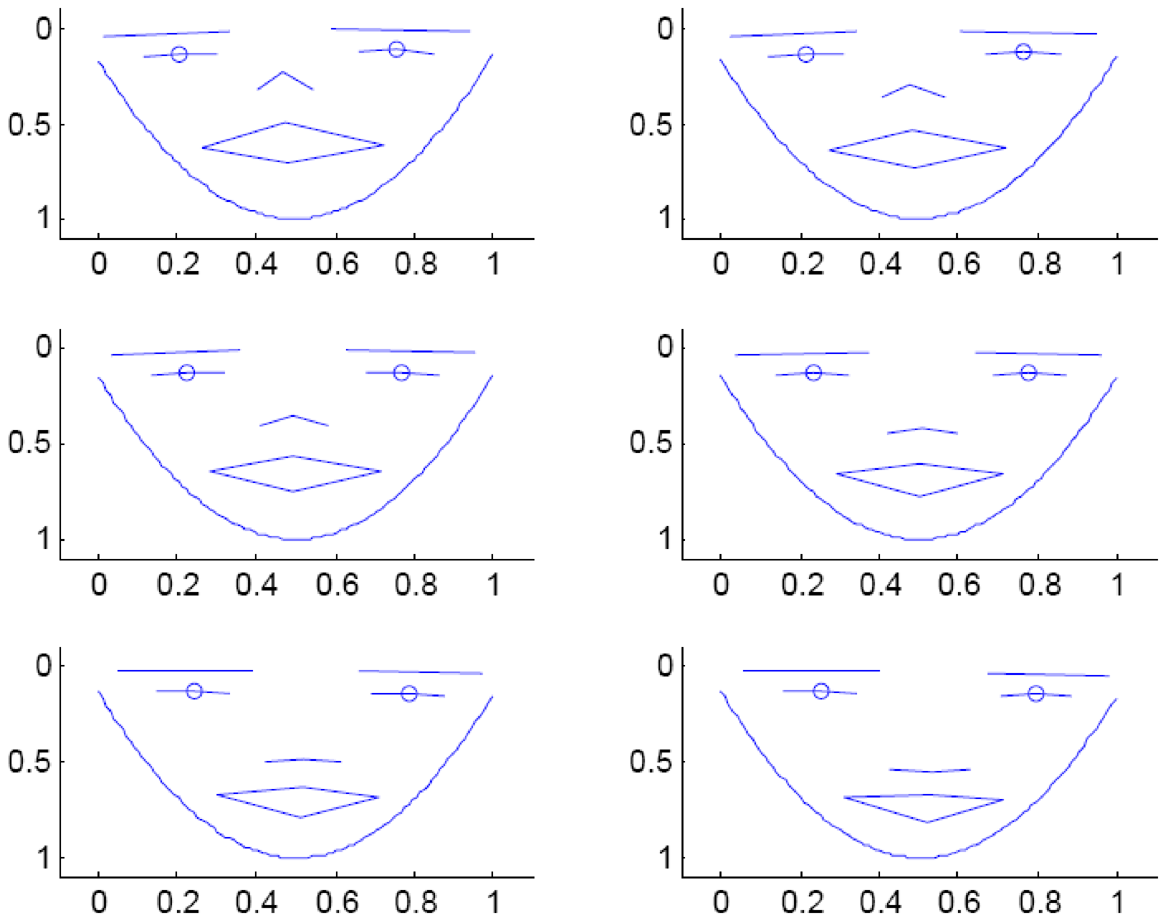
ASMs for Human Face Images

- Effect of varying first PCA feature



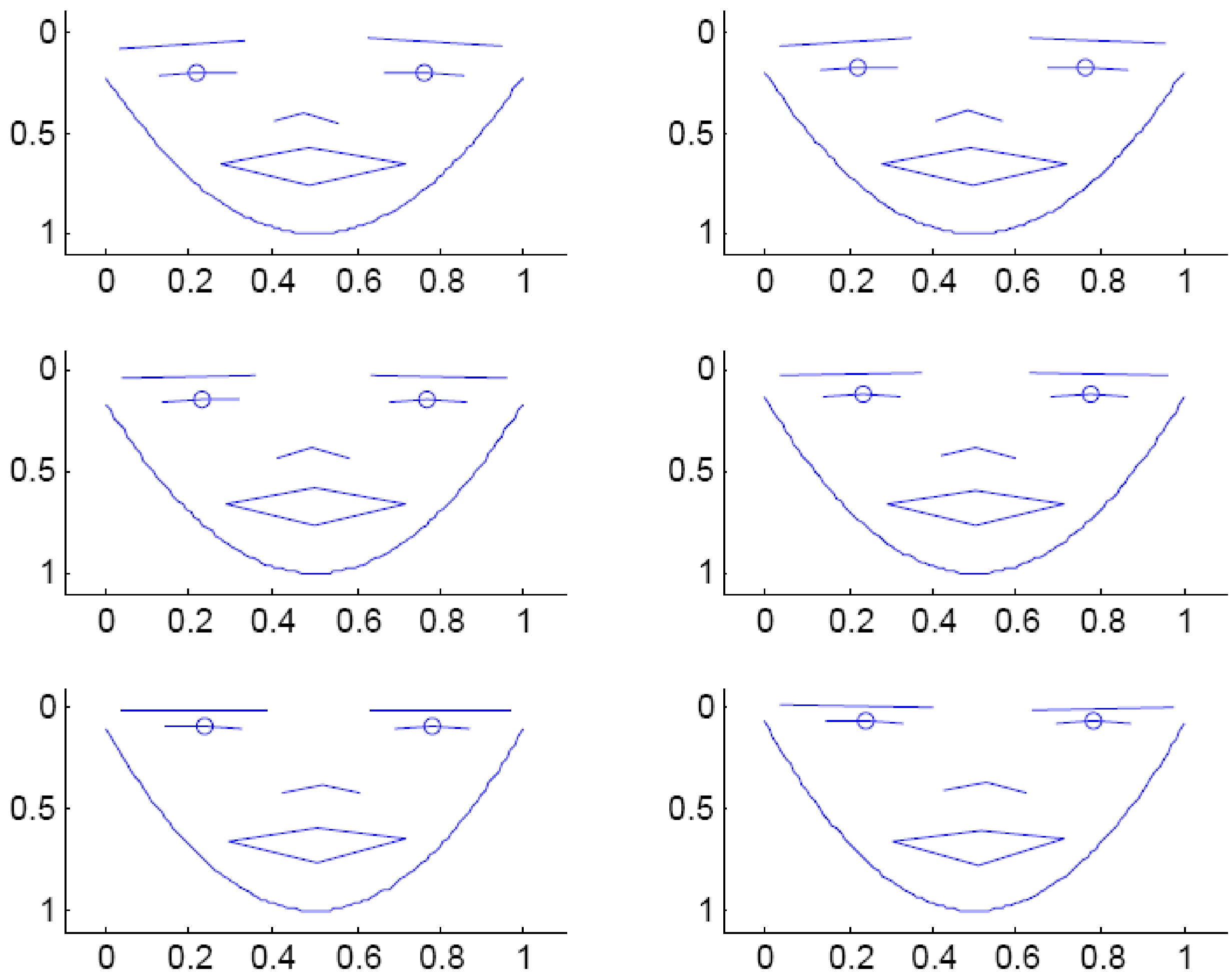
ASMs for Human Face Images

- Effect of varying second PCA feature



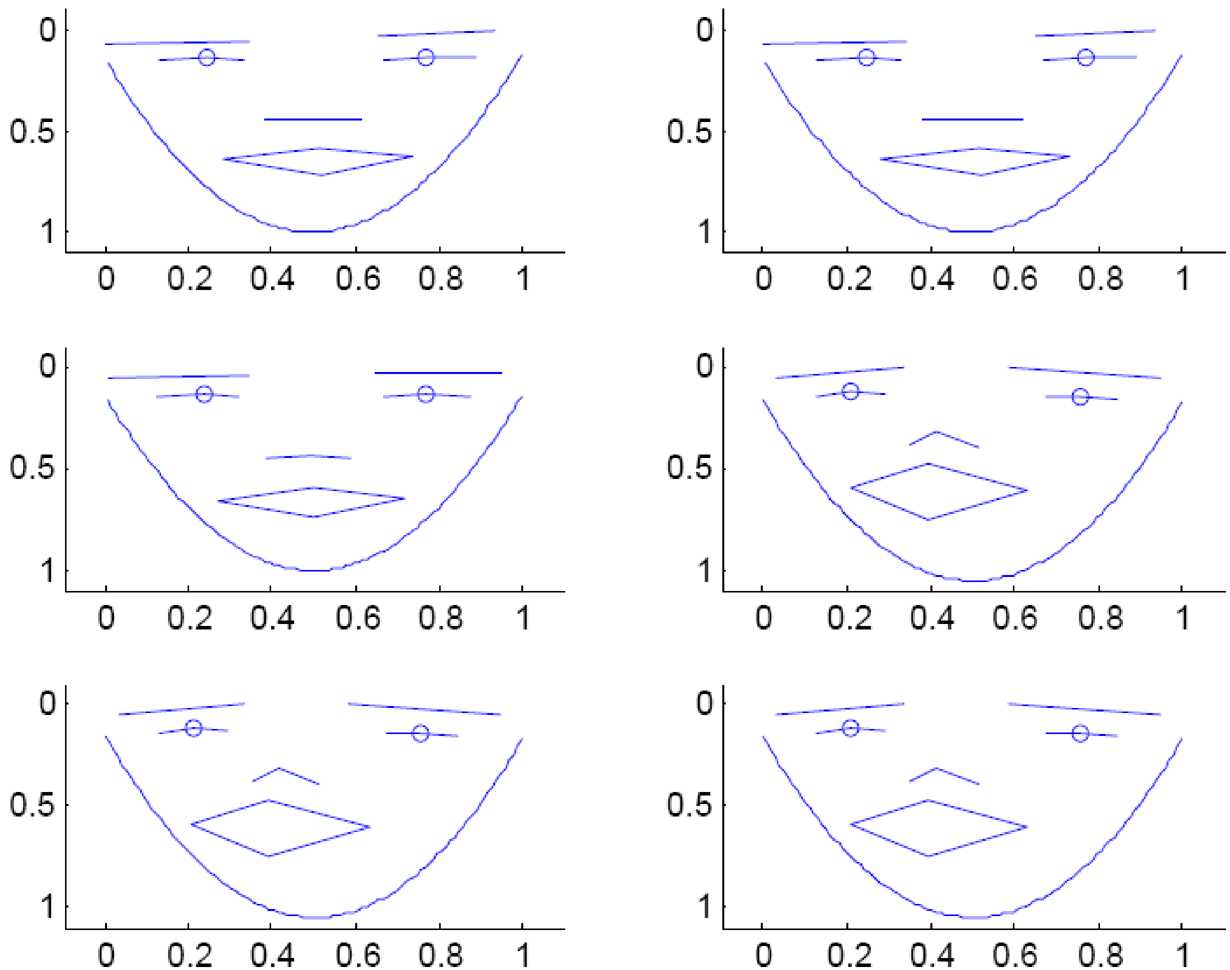
ASMs for Human Face Images

- Effect of varying third PCA feature



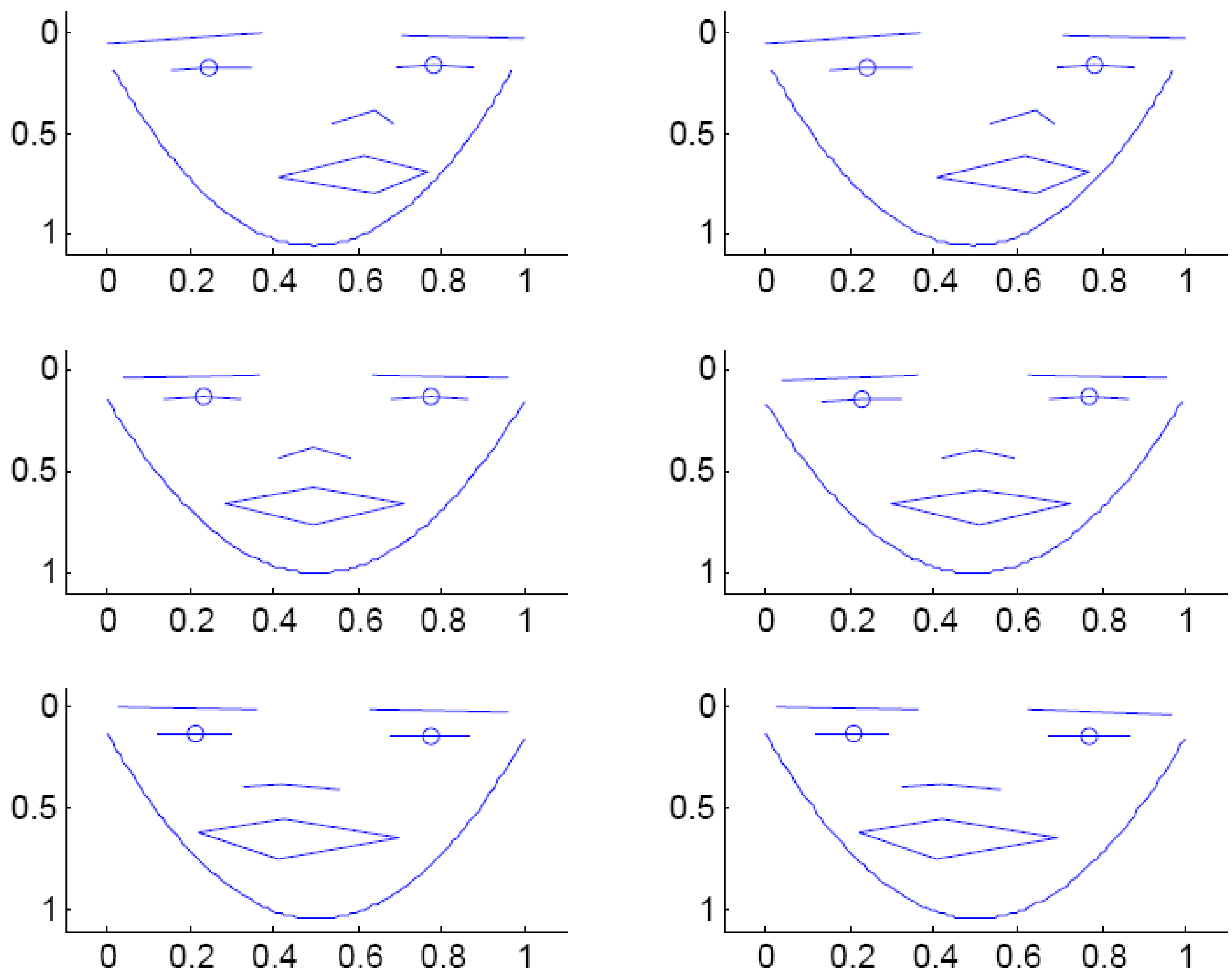
ASMs for Human Face Images

- Effect of varying first kernel PCA feature



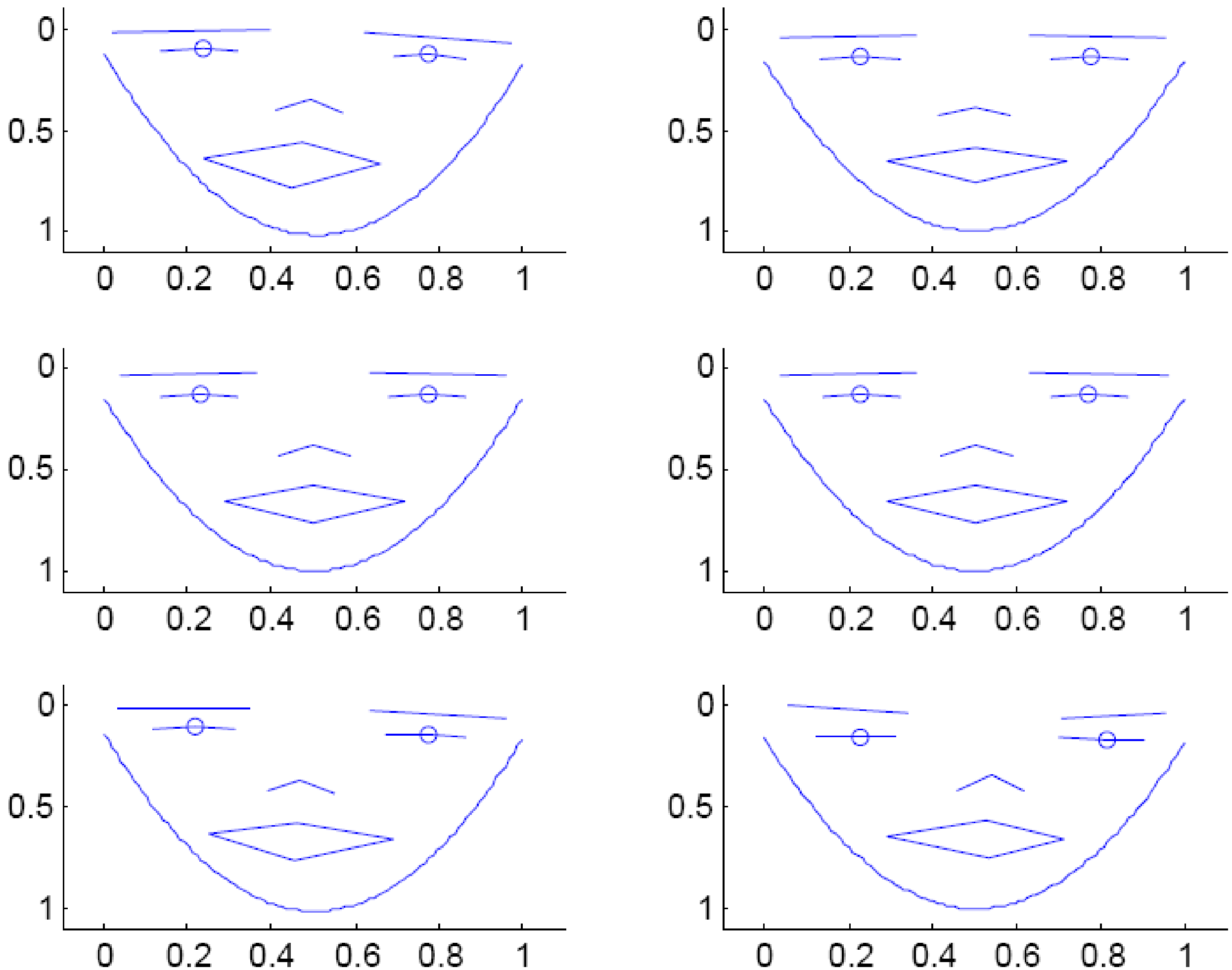
ASMs for Human Face Images

- Effect of varying second kernel PCA feature



ASMs for Human Face Images

- Effect of varying third kernel PCA feature



ASMs for Human Face Images

- Gaussian kernel PCA based ASMs is promising to
  - Discover hidden deformation patterns that is not revealed by standard ASMs
  - Recognize microexpressions that cannot be represented in a linear subspace of original feature space

Parameter Selection

- Gaussian kernel relies on the value of the distance between two vectors $\|\mathbf{x} - \mathbf{y}\|$
- σ : Determine the capture range of the kernel function. If it is too small, kernel method will be invalid.
- We define:

$$MedMin = \text{median}_n \min_{m \neq n} \|\mathbf{x}_n - \mathbf{x}_m\|$$

$$\sigma = 10 \times MedMin$$

Summary

- Studied the theories of kernel PCA and Active Shape Models (ASMs)
- Implemented kernel PCA and used it to design nonlinear ASMs
- Tested kernel PCA for
  - Synthetic data classification
  - Human face images classification
  - Building nonlinear ASMs for human face models
- Designed a parameter selection method