Outline

- Introduction
- Principal Components Analysis (PCA)
- Eigenface Algorithm
- Image Retrieval and Preprocessing
- Feature Detection
- Feature-Based Age Progression
- Conclusion
Introduction

- Project attempts to age digital images of faces
- Application of PCA on image data
- Two-Phase Process
  - Training: compute lower dimension coordinate system for data
  - Reconstruction: project input image onto new coordinate system to obtain weight vector; reconstruct image with weight vector
- Training data - concatenated (young, aged) face pairs
- Want projected input image to be near cluster of projected training images with desired aged features
- Reconstruct input image to capture aged features of cluster by a weighted averaging effect
Principal Components Analysis (PCA)

- Transform coordinate system of data set so new axes in directions of max. scatter
- Axes with little point spread truncated so fewer variables needed to represent data
- Compute eigenvectors/eigenvalues from covariance matrix
- Principal components: eigenvectors with top eigenvalues
- Lossy process as some information is lost (e.g. spread information about $e_2$)
Eigenface Algorithm

- Face class - group of images of same person
- $\Omega_k$ - average weight vector in face class $k$
- $\varepsilon_k$ - smallest between input weight vector and average weight vector for all face classes
Eigenface Program

- C++ and MFC implementation of eigenface algorithm
- Performs image recognition and reconstruction
- Supports only grayscale images
Image Retrieval and Preprocessing

- Consists of three Python command line scripts and executable
- `webcrawler.py` interacts with www.missingkids.com to retrieve (young, aged) pairs
- Uses face detection API [8] to locate eyes and face
Reconstruction Test

- Use scripts to retrieve and preprocess 300 female and 200 male (young, aged) grayscale face pairs (100 by 100 pixels)
- Assign 200 female pairs as training images
- Assign 150 male pairs as training images
- Separately reconstruct remaining female and male input images
Reconstruction Test Results

- Poor reconstruction and hence poor aging results stem from inadequately sized images
- Smaller images have less descriptive (lower-dimensionality) principal components
- Less descriptive principal components mean less accurate reconstruction

<table>
<thead>
<tr>
<th></th>
<th>Female</th>
<th>Male</th>
</tr>
</thead>
<tbody>
<tr>
<td>Successful reconstruction</td>
<td>60 / 100</td>
<td>30 / 50</td>
</tr>
<tr>
<td>Successful aged</td>
<td>15 / 100</td>
<td>10 / 50</td>
</tr>
</tbody>
</table>

Some successful reconstructed aged faces from test
Recognition Test

- Downloaded 400 (100 by 100 pixels) grayscale images from [http://www.uk.research.att.com/facedatabase.html](http://www.uk.research.att.com/facedatabase.html)
- Images normalized in terms of lighting, cropping of head, and background removal
- Separate images into 40 face classes with 5 images each person for training
- Remaining 200 images for testing
Recognition Test Results

- Classified the 200 test images and obtained:
  - Correctly classified faces - 92%
  - Incorrectly classified faces - 8%
  - Minimum $\varepsilon_{\text{class}}$ - 1151.56
  - Maximum $\varepsilon_{\text{class}}$ - 3601.87

- Set $\theta_{\text{class}}$ larger than the max. $\varepsilon_{\text{class}}$ value

- Maximizes % of correctly classified faces at expense of more incorrectly classified faces
Extension of Eigenface Program

- Support reconstruction of color face images
- Feature-based approach to age progression
  - Locate face and major features on face
  - Age progress individual features
  - Blend aged features into aged face
- First attempt to locate features uses shape context descriptor method
Shape Context Descriptor

- Edge detect and sample N points from edges
- For a point, form vectors to all other N – 1 points; N * (N – 1) total vectors
- Use log-polar histogram to sort vectors

Assign $\chi^2$ cost

$$C_{ij} = \frac{1}{2} \sum_{k=1}^{K} \frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)}$$

- $h_i(k)$ represents bin k for point i
- N by N matrix to represent costs of all point pairs between two shapes
Shape Context Descriptor (continued)

- Solve cost matrix as constraint optimization problem
- Select groups of points for features on one shape
- Find corresponding points on second shape
- Determine centroid of each group on second shape to approximately locate features
- Remove outliers greater than a number of standard deviations from the mean radial distance between the group center and each point
Feature Matching Program

- C++ and MFC implementation of shape context descriptors
- Features on one image to be matched to another image
- May tweak number of sampled points, bin count, and edge detection sensitivity to improve performance
Feature Matching Test

- Three test sets of shapes and faces
- For shapes, 36 pairs of images of alphanumeric characters (A – Z, 0 – 9)
- Retrieve images from www.missingkids.com
- Set of 50 (young, aged) image pairs
- Set of 50 pairs of different faces
# Feature Matching Results

<table>
<thead>
<tr>
<th></th>
<th>Successful Feature Matching</th>
</tr>
</thead>
<tbody>
<tr>
<td>Character images</td>
<td>32 / 36</td>
</tr>
<tr>
<td>(young, aged) faces</td>
<td>30 / 50</td>
</tr>
<tr>
<td>Different faces</td>
<td>15 / 50</td>
</tr>
</tbody>
</table>

- Examples of successful matching on the right
- Poor results for different faces
- Conclusion: need more robust face / feature location method
Feature-Based Age Progression

- Young image
- Aged image
  - Extract and separate RGB values
    - Red image
    - Green image
    - Blue image
  - Red image
  - Green image
  - Blue image

- Young faces
- Aged faces
  - Locate features and extract (eyes, nose, mouth, and face)
  - Resize corresponding features to common dimensions
    - Young feature image (H by W)
    - Aged feature image (H by W)
  - Extract and separate RGB values
    - Form (3H by 2W) training feature image matrix
    - N encoded matrices for each feature
  - Age progression
    - Compute principal components for each feature
    - Project input feature onto corresponding eigenfeature
    - Weight vector
    - Reconstruct aged feature
    - Aged-progressed features
  - Blend aged features and form overall aged face

- Input color face image
Feature-Based Age Progression (continued)

- Uses neural net based face detection API [8] to locate eyes and face boundary
- Use distance between eyes as metric to locate and bound other features
- Dimension scheme gives rough bounding boxes only
- Tighten bounding boxes using edge detection data
Feature-Based Age Progression (continued)

- Input features and principal components of each feature (eigenfeatures)
- Age-progress face and individual features
- Find best-fit contour around each feature
- Blend aged features back into face
- Result should be smooth seamless aged face
Feature-Based Program

- C++ and MFC implementation
- Automatically extracts features, or manually select features to train or age
- Improve results with various parameters:
  - Edge detection parameter for tighter bounding boxes
  - Increase low-pass filter value for better blending of features
Feature-Based Test

- Use scripts to retrieve and prepare 300 female and 200 male (young, aged) color images (240 by 300 pixels)
- Train with 200 of the 300 female image pairs
- Reconstruct remaining young female images
- Train with 150 of the 200 male image pairs
- Reconstruct remaining young male images
Feature-Based Test Results

- Improvement over previous test with grayscale eigenface program
  - Improvement in % of reconstructions
  - 100% and 50% improvement for female and male aging results, respectively
- Attributed to larger sized training images and use of color

<table>
<thead>
<tr>
<th></th>
<th>Female</th>
<th>Male</th>
</tr>
</thead>
<tbody>
<tr>
<td>Successful reconstruction</td>
<td>85 / 100</td>
<td>35 / 50</td>
</tr>
<tr>
<td>Successful aging</td>
<td>30 / 100</td>
<td>15 / 50</td>
</tr>
</tbody>
</table>

Example feature-based aged results
Clustering Test

- Create training images of all pairs between baby, adolescent, and toddler images with adult images (images to the right)
- Total of $(6 + 6 + 4) \times 6 = 96$ training images to cluster images
- First train and test with the 96 images only
- Then add 100 images of other people to see how clustering performs with mixed training set
Clustering Results

- Test images on top row
- Mid row results of 96 training images
- Lower row results of extended training image set
- Row illustrates a weighted average effect from all training images
- Additional training images skew reconstructed results
Colorization Test

- Concatenate training image pairs to consist of a gray and color image of the same person
- Convert application from one that ages a color face image, to one that colorizes a grayscale face image
- Use scripts to retrieve 300 (240 by 300 pixels) color face images from www.missingkids.com
- Convert 200 images to grayscale and concatenate color to grayscale images to form training image set
- Colorize 100 grayscale test images
Colorization Results

- 90 of the 100 test images are reasonably colorized with consistent skin tone
- Images tend to exhibit areas of gray blending into color
- Reconstructed faces vary slightly with input gray faces
Conclusion

- Clustering test summary:
  - Results highly sensitive to inclusion of other training images
  - Reconstruction captures spurious features from training images and skews projection from intended cluster of training images
  - Proposed solution: reconstruct image from average weight vector of a matched face class using $\varepsilon_k$ metric

- Runtime bottleneck during training
  - Computation of eigenvectors and eigenvalues, and formation of principal components
  - RGB image encoding increases runtime by a factor of six
  - Need a more efficient color encoding in terms of matrix size

- Improve current feature extraction method
  - Poor for out-of-plane face rotations
  - Poor for faces with atypical face feature ratios
  - Need more robust extraction method (e.g. image segmentation, customized neural application)
References