CSE 402 - BIOMETRICS AND PATTERN RECOGNITION

PROF. ARUN ROSS

MICHIGAN STATE UNIVERSITY

THANKS TO: THOMAS SWEARINGEN

FACE RECOGNITION PART 2

TERMINOLOGY

- **Label**: a specific class of a data sample (e.g., an image of a face is labeled with an ID of the person)
- **Training Data**: labeled examples used by a classifier to learn a task
- **Test Data**: data (*separate from training data*) used to evaluate classifier performance; a test label is compared with the classifier predicted label to evaluate classifier performance
- **Supervised Learning Method**: a method that uses class labels of the training data to learn
- **Unsupervised Learning Method**: a method that does not use the class labels of the training data to learn

SOME APPROACHES TO FACE MATCHING

- Appearance-Based
  - **Principal Component Analysis**
  - **Linear Discriminant Analysis**
- Model-Based
  - Elastic Bunch Graph Matching
- Texture-Based
  - Local Binary Pattern (LBP)
- Deep Learning
  - Convolutional Neural Network

FACE RECOGNITION

PRINCIPAL COMPONENT ANALYSIS (PCA)

PRINCIPAL COMPONENT ANALYSIS (PCA)

- Early automated face recognition method
- **Goal:** learn a subspace that accounts for as much variability in the training data as possible
- Does not use identity information during training (*Unsupervised Learning*)

PCA DATA SETUP

$d \times 1$ column vector

1. Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N$ denote each training sample
2. Compute average of the training set

$d \times 1$ column vector

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_i$$

3. Define the data matrix

$d \times N$ matrix

$$X = \left[(\boldsymbol{x}_1 - \boldsymbol{\mu}) \; (\boldsymbol{x}_2 - \boldsymbol{\mu}) \cdots (\boldsymbol{x}_N - \boldsymbol{\mu}) \right]$$

PCA SUBSPACE LEARNING

4. Calculate the covariance matrix

$$C = XX^T$$

$d \times d$ matrix

$d \times N$ matrix $N \times d$ matrix

5. Compute the Eigenvectors of the covariance matrix by solving

$d \times 1$ column vector

$$CE = \lambda E$$

where $E = [e_1, e_2, \ldots, e_d]$ are the Eigenvectors

$d \times d$ matrix

PCA COMPARING FACES

Instead of taking all "d" eigen-vectors, Ek takes the top "k" eigen-vectors (eigen-vectors are arranged in columns)

1. Represent two unseen images, $\mathbf{x_i}$ and $\mathbf{x_j}$, as a weighted sum of Eigenvectors

$k \times 1$ vector $\longrightarrow$ $$\boldsymbol{\omega}_i = \boldsymbol{E}_k{}^T(\boldsymbol{x}_i - \boldsymbol{\mu}) \quad \boldsymbol{\omega}_j = \boldsymbol{E}_k{}^T\left(\boldsymbol{x}_j - \boldsymbol{\mu}\right)$$

2. Compare difference of weighted sums to threshold (t)

$$\| \boldsymbol{\omega}_i - \boldsymbol{\omega}_j \|^2 \lesseqgtr t$$

Faces are considered a match if the difference is less than the threshold, otherwise they are not a match

Algorithm 1 Principal Component Analysis for Faces

$\boldsymbol{X}_{\text{train}} \leftarrow \text{LoadTrainData}()$ ▷ $\boldsymbol{X}_{\text{train}} = \left[\boldsymbol{x}_1^{\text{train}}, \ldots, \boldsymbol{x}_N^{\text{train}}\right]$
$\boldsymbol{X}_{\text{test}} \leftarrow \text{LoadTestData}()$ ▷ $\boldsymbol{X}_{\text{test}} = [\boldsymbol{x}_1^{\text{test}}, \ldots, \boldsymbol{x}_M^{\text{test}}]$
$\boldsymbol{V}, \boldsymbol{\mu} \leftarrow \text{Eigenfaces}\,(\boldsymbol{X}_{\text{train}})$ ▷ Learn Subspace
$\boldsymbol{D} \leftarrow \text{CompareFaces}\,(\boldsymbol{X}_{\text{test}}, \boldsymbol{V}, \boldsymbol{\mu})$ ▷ Obtain distances scores for test data

function EigenFaces($\boldsymbol{X}$)
  $\boldsymbol{\mu} \leftarrow \frac{1}{N}\sum_{I=1}^{N} \boldsymbol{x}_i$ ▷ calculate mean
  $\boldsymbol{X}_{\text{c}} \leftarrow \boldsymbol{X} - [\boldsymbol{\mu}, \ldots, \boldsymbol{\mu}]$ ▷ center data
  $\boldsymbol{C} \leftarrow \boldsymbol{X}_{\text{c}}\boldsymbol{X}_{\text{c}}^{\text{T}}$ ▷ compute covariance matrix
  $\boldsymbol{V} \leftarrow \text{Eigenvectors}(\boldsymbol{C})$ ▷ Find Eigenvectors
  return $\boldsymbol{V}, \boldsymbol{\mu}$
end function

function CompareFaces($\boldsymbol{X}, \boldsymbol{V}, \boldsymbol{\mu}$)
  $\boldsymbol{\Omega} = \boldsymbol{V}^{\text{T}}\,(\boldsymbol{X} - [\boldsymbol{\mu}, \ldots, \boldsymbol{\mu}])$ ▷ $\boldsymbol{\Omega} = [\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_M]$
  for each $\boldsymbol{\omega}_i, \boldsymbol{\omega}_j$ pair in $\boldsymbol{\Omega}$ **do**
    $D_{i,j} = \|\boldsymbol{\omega}_i - \boldsymbol{\omega}_j\|^2$ ▷ Compare test samples
  end for
  return $\boldsymbol{D}$
end function

VISUALIZING PCA EIGENVECTORS

- Train on **4,000 aligned images** from the *Oxford VGG Face dataset*
- Obtain the Eigenvectors corresponding to the 4 largest Eigenvalues
  - Called *Eigenfaces*
- Reshape from 1D vector to 2D image

Mean

Largest Eigenvalue

2nd Largest Eigenvalue

3rd Largest Eigenvalue

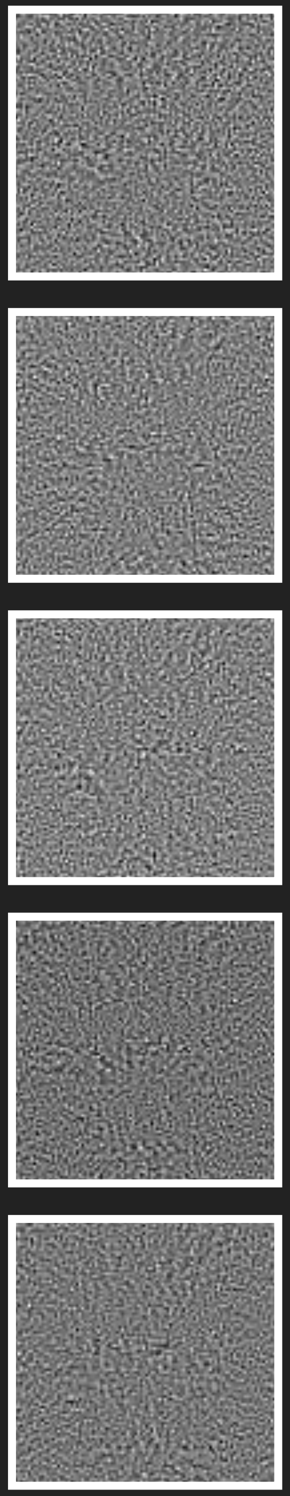
4th Largest Eigenvalue

COMPRESSED REPRESENTATION

- **Not all Eigen vectors needed to adequately represent the data**
- Eigenvectors with small Eigenvalues **mostly model noise** rather than discriminant info relating to identity
- Represent data using only k Eigenvectors $(k < d)$

$k \times 1$ vector $\longrightarrow$ $\boldsymbol{\omega}_i = \boldsymbol{E}_k^T \boldsymbol{x}$

E_k is $d \times k$



5th Smallest Eigenvalue

4th Smallest Eigenvalue

3rd Smallest Eigenvalue

2nd Smallest Eigenvalue

Smallest Eigenvalue

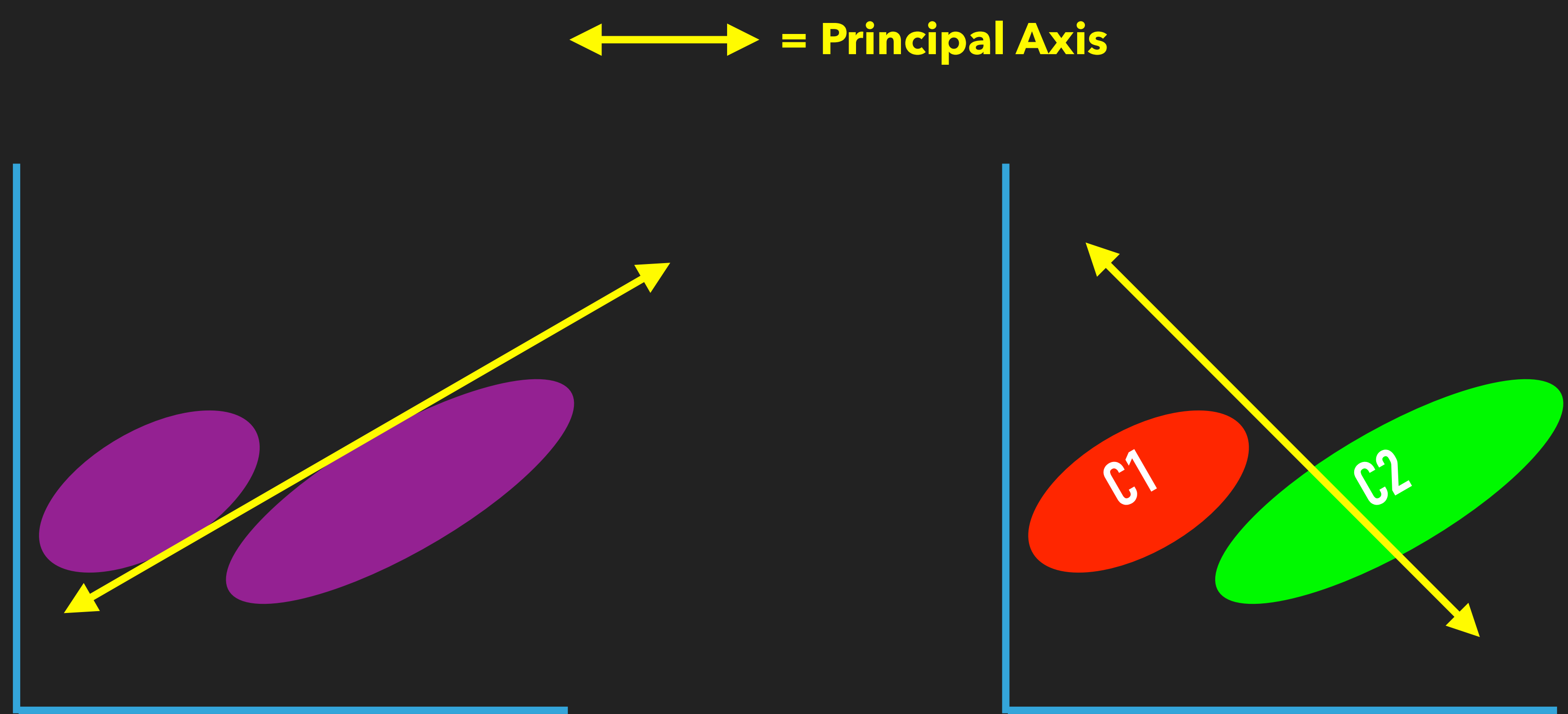
FACE RECOGNITION

LINEAR DISCRIMINANT ANALYSIS (LDA)

LINEAR DISCRIMINANT ANALYSIS (LDA)

- Unlike PCA, LDA makes use of identity information (*Supervised Learning*)
- **Goal:** learn a subspace that *minimizes intra-class variation* and maximizes inter-class variation
  - Images from the same subject are closer together
  - Images from different subjects further apart

PCA VS LDA



PCA

Principal axis follows the direction with the most variation for all data.

LDA

Principal axis follows the direction with the most *inter-class* variation.

Illustration adapted from *Introduction to Biometrics*, 2011.

LDA INITIALIZATION

1. Let $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)$ be the training set where $\boldsymbol{x}_i$ is the data and $y_i = \{1, \ldots, C\}$ is the class label

2. Compute mean for each class (identity)

$d \times 1$ column vector

$$\boldsymbol{\mu}_c = \frac{1}{N_c} \sum_{\{i \mid y_i = c\}} \boldsymbol{x}_i$$

where N_c is the number of samples with class c

LDA SCATTER MATRICES

3. Use training data to calculate between class and within class scatter matrices

within class → $d \times d$ matrix

$$\boldsymbol{S}_w = \sum_{c=1}^{C} \sum_{\{i \mid y_i = c\}} \left(\boldsymbol{x}_i - \boldsymbol{\mu}_c\right)\left(\boldsymbol{x}_i - \boldsymbol{\mu}_c\right)^T$$

between class → $d \times d$ matrix

$$\boldsymbol{S}_b = \sum_{c=1}^{C} N_c \left(\boldsymbol{\mu}_c - \boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_c - \boldsymbol{\mu}\right)^T$$

where $\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i$ (mean of all data)

LDA SUBSPACE

4. Calculate Eigenvectors by solving

minimized → $$S_w^{-1} S_b E = \lambda E$$ ← maximized

5. Represent any data vector $\boldsymbol{x}$ by

$d \times 1$ vector → $$\boldsymbol{\omega}_i = \boldsymbol{E}^T \left(\boldsymbol{x} - \boldsymbol{\mu}\right)$$

6. Compare representations like in PCA (norm of difference)

Algorithm 1 Linear Discriminant Analysis for Faces

$\boldsymbol{X}_{\text{train}}, \boldsymbol{y}_{\text{train}} \leftarrow \text{LoadTrainData}()$ ▷ $\boldsymbol{y}_{\text{train}} = [y_1^{\text{train}}, \ldots, y_N^{\text{train}}]$
$\boldsymbol{X}_{\text{test}} \leftarrow \text{LoadTestData}()$
$\boldsymbol{V}, \boldsymbol{\mu} \leftarrow \text{Fisherfaces}(\boldsymbol{X}_{\text{train}}, \boldsymbol{y}_{\text{train}})$ ▷ learn subspace
$\boldsymbol{D} \leftarrow \text{CompareFaces}(\boldsymbol{X}_{\text{test}}, \boldsymbol{V}, \boldsymbol{\mu})$ ▷ obtain distances scores for test data

function FisherFaces($\boldsymbol{X}, \boldsymbol{y}$)
    for each $c \in [1, \ldots, C]$ **do**
        $N_c \leftarrow |\{i | y_i = c\}|$ ▷ num. samples with class c
        $\boldsymbol{\mu}_c \leftarrow \frac{1}{N_c} \sum_{\{i|y_i=c\}} \boldsymbol{x}_i$ ▷ calculate class mean
    end for
    $\boldsymbol{\mu} \leftarrow \frac{1}{N} \sum_{i=1}^{N} \boldsymbol{x}_i$ ▷ calculate overall mean
    $\boldsymbol{S}_w = \sum_{c=1}^{C} \sum_{\{i \mid y_i=c\}} (\boldsymbol{x}_i - \boldsymbol{\mu}_c)(\boldsymbol{x}_i - \boldsymbol{\mu}_c)^{\mathrm{T}}$
    $\boldsymbol{S}_b = \sum_{c=1}^{C} N_c (\boldsymbol{\mu}_c - \boldsymbol{\mu})(\boldsymbol{\mu}_c - \boldsymbol{\mu})^{\mathrm{T}}$
    $\boldsymbol{V} \leftarrow \text{Eigenvectors}(\boldsymbol{S}_w^{-1} \boldsymbol{S}_b)$ ▷ find Eigenvectors
    return $\boldsymbol{V}, \boldsymbol{\mu}$
end function

function CompareFaces($\boldsymbol{X}, \boldsymbol{V}, \boldsymbol{\mu}$)
    $\boldsymbol{\Omega} = \boldsymbol{V}^{\mathrm{T}} (\boldsymbol{X} - [\boldsymbol{\mu}, \ldots, \boldsymbol{\mu}])$ ▷ $\boldsymbol{\Omega} = [\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_M]$
    for each $\boldsymbol{\omega}_i, \boldsymbol{\omega}_j$ pair in $\boldsymbol{\Omega}$ **do**
        $D_{i,j} = \|\boldsymbol{\omega}_i - \boldsymbol{\omega}_j\|^2$ ▷ compare test samples
    end for
    return $\boldsymbol{D}$
end function

VISUALIZING LDA EIGENVECTORS

- Train on images from the *ORL Face dataset*
- Obtain the Eigenvectors corresponding to the 5 largest Eigenvalues
  - Called *Fisherfaces*
- Reshape from 1D vector to 2D image

Largest Eigenvalue

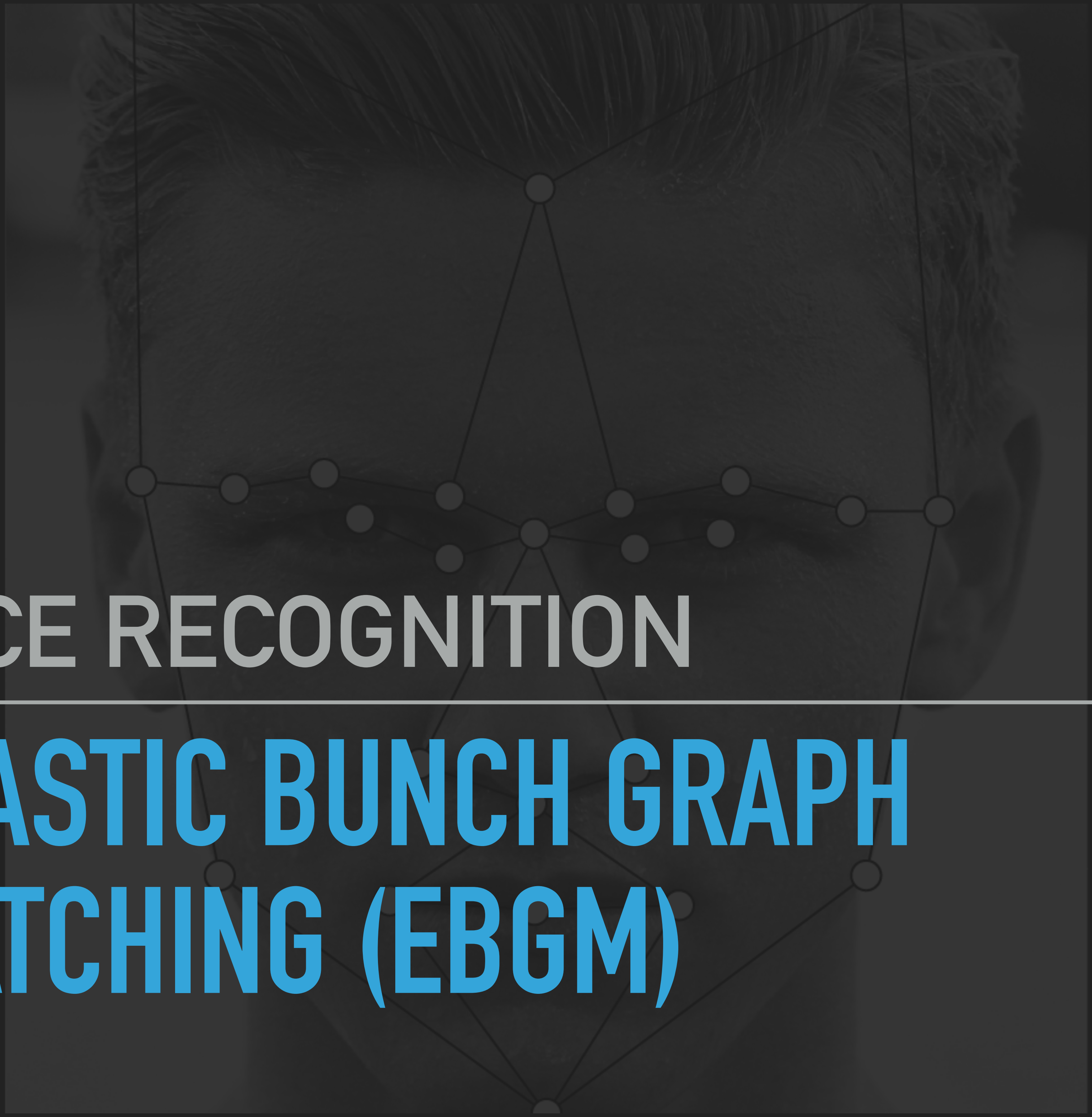
2nd Largest Eigenvalue

3rd Largest Eigenvalue

4th Largest Eigenvalue

5th Largest Eigenvalue

Images from *Introduction to Biometrics*, 2011.



FACE RECOGNITION

ELASTIC BUNCH GRAPH MATCHING (EBGM)

ELASTIC BUNCH GRAPH MATCHING (EBGM)

- Model-based method
- Robust to *occlusions* or *pose*
- Uses ***Gabor filters at specific face landmarks***
- Organizes filter responses in a **graph**

FACE LANDMARKS

- Point of interest on the face
  - corners of eyes
  - tip of the nose
  - corners of the mouth
  - etc.

- Also known as fiducial points

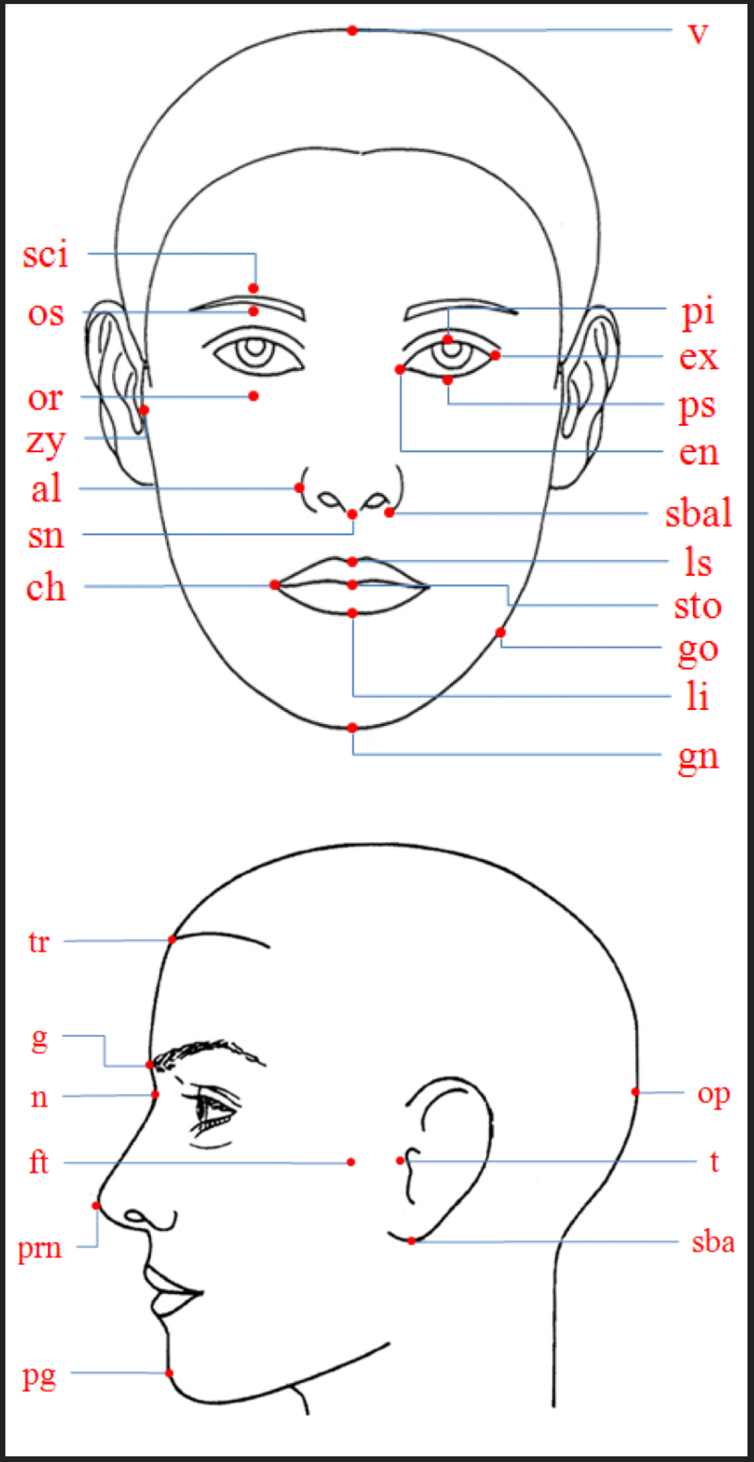
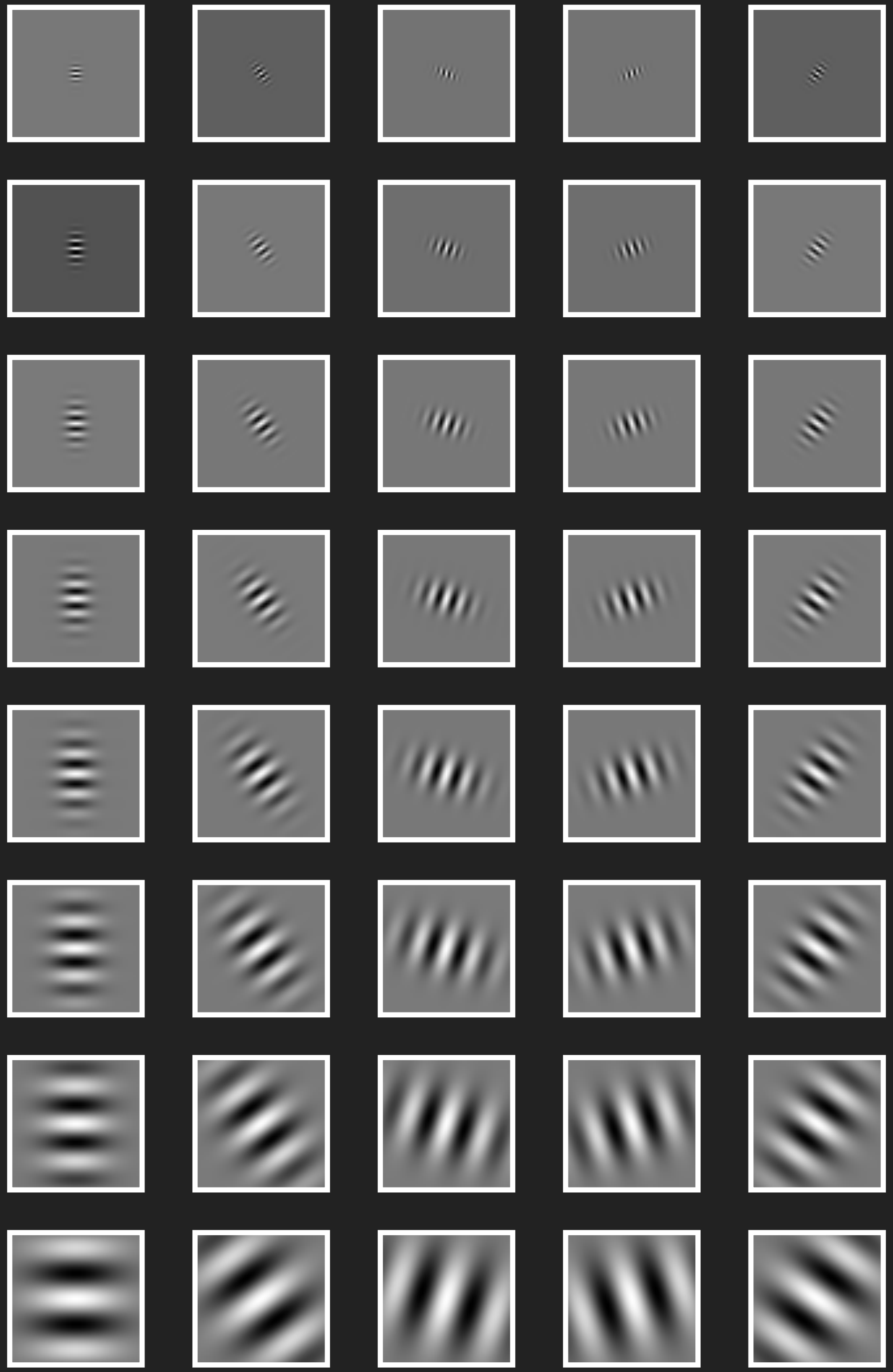


Image from *Introduction to Biometrics*, 2011.
(adapted from *Anthropometry of the Head and Face*, 1994)

GABOR FILTERS

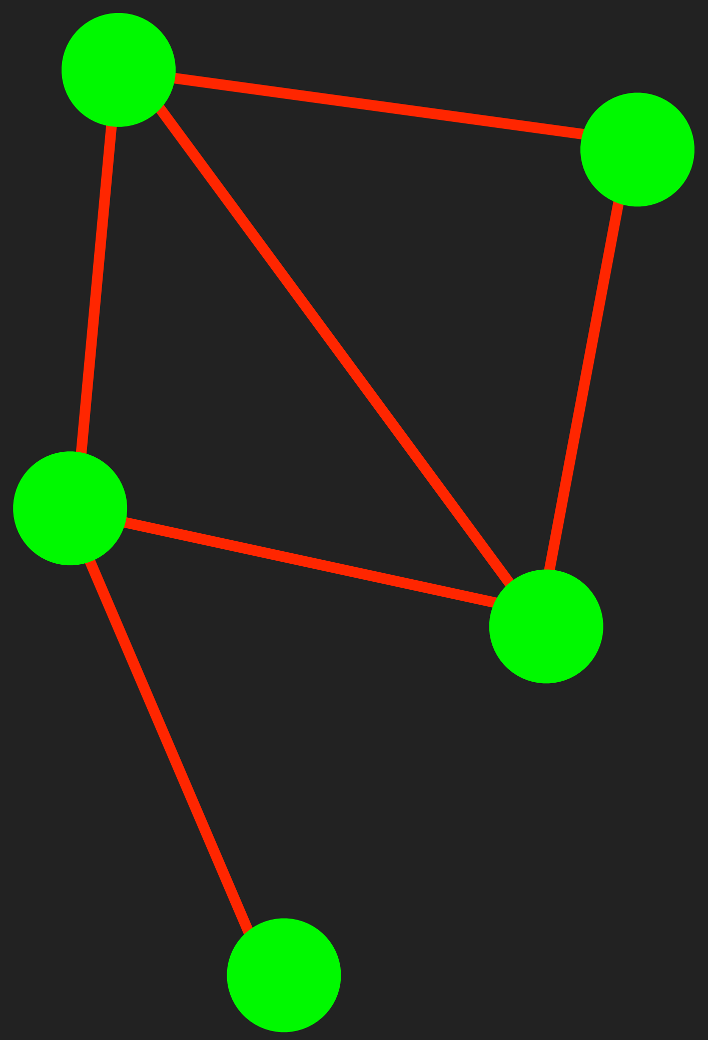


- Apply a set of filters to a *small area* centered around each landmark
- Total of 40 responses for each landmark
  - 8 different scales
  - 5 different directions
- A set of filter responses is called a **jet**

Images produced using code from https://stackoverflow.com/questions/16255068/how-to-show-40-gabor-filter-in-matlab

GRAPHS

- Structure consisting of nodes and edges
- Edges relate nodes to one another
- Edges may have an associated weight that defines a relationship between 2 nodes



TRAINING – 1ST STAGE

- Landmark points are identified *manually* for the first few images
- Landmarks for subsequent images are discovered *automatically* by comparing the Gabor jets
- Manual correction of mislabeled landmarks may be required
- Edge weights denote the distance between face landmarks

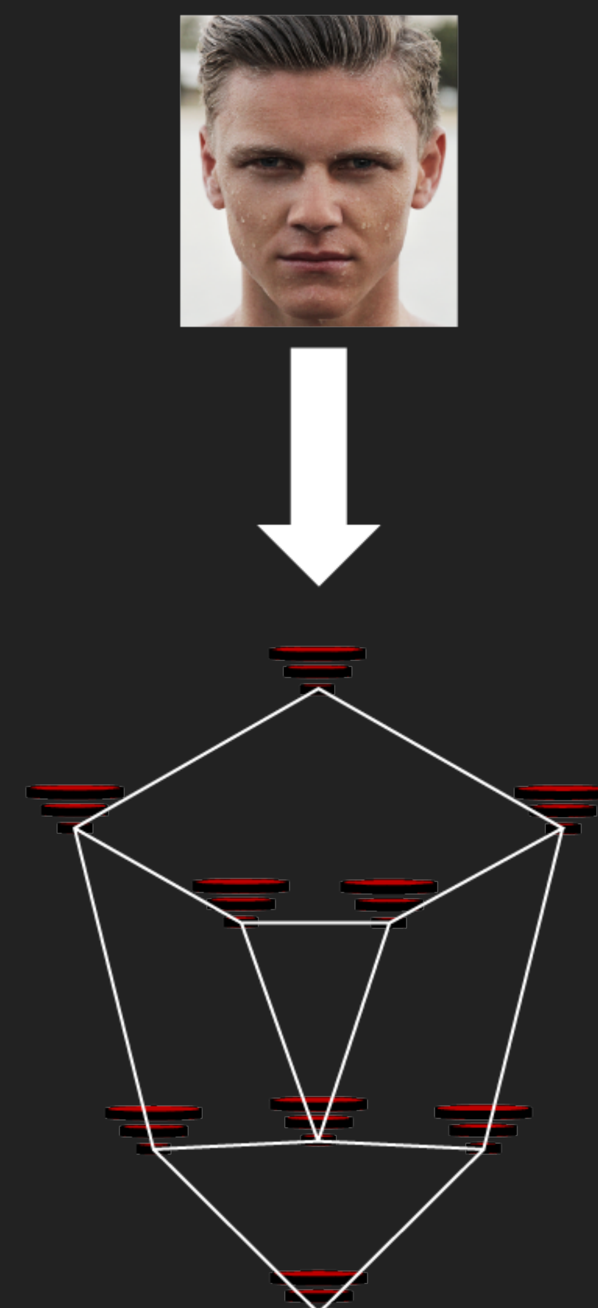


Illustration adapted from *Introduction to Biometrics*, 2011.

TRAINING – 2ND STAGE

- Image graphs are combined into a stack-like structure called a Face Bunch Graph (*FBG*)
- The combined jets for a particular landmark is called a ***bunch***
- Several FBGs may be created for different poses

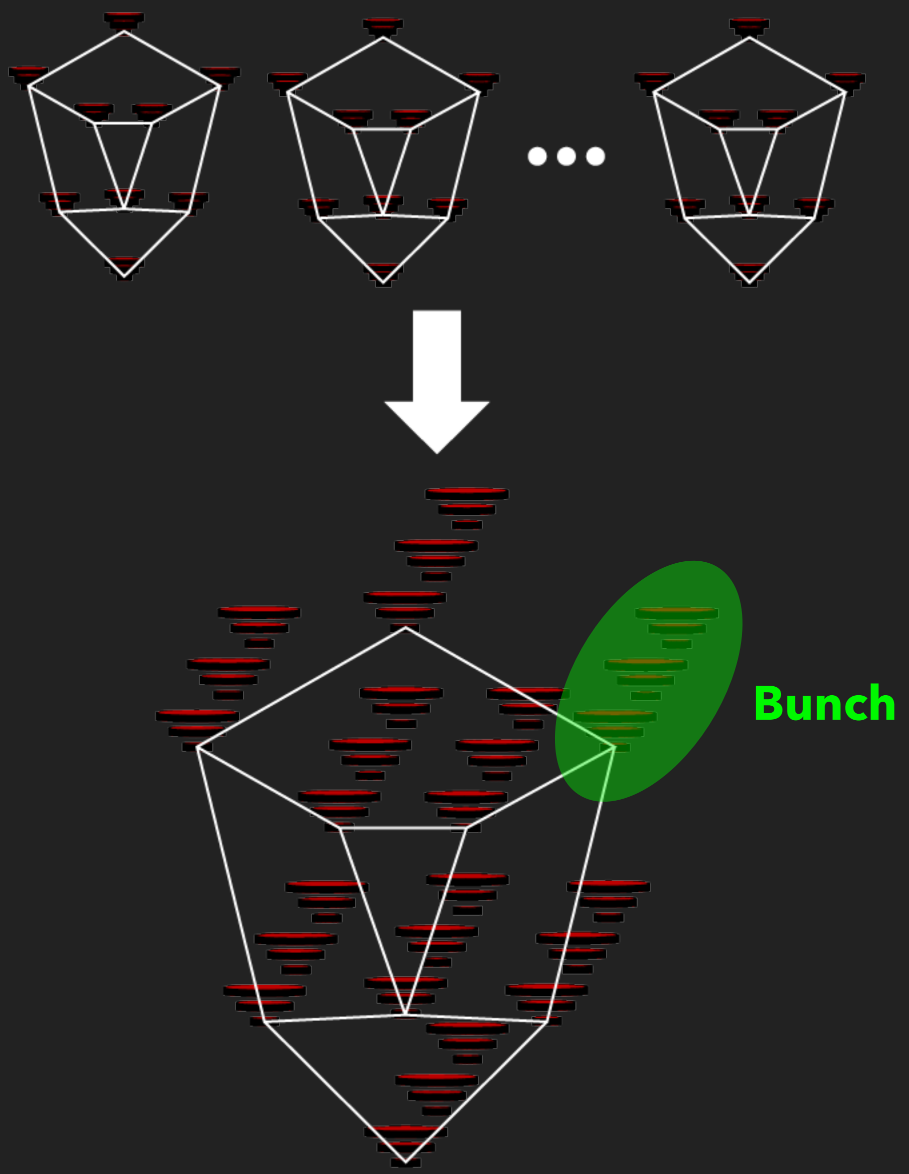


Illustration adapted from *Introduction to Biometrics*, 2011.

MATCHING USING FBG

- Obtain approximate face position for an image using a condensed FBG (each bunch is averaged)
- Refine face position using the full FBG
- Result is a fitted graph for an image
- Fitted graphs for two face images are compared by taking the **average similarity between the jets at each corresponding landmark**

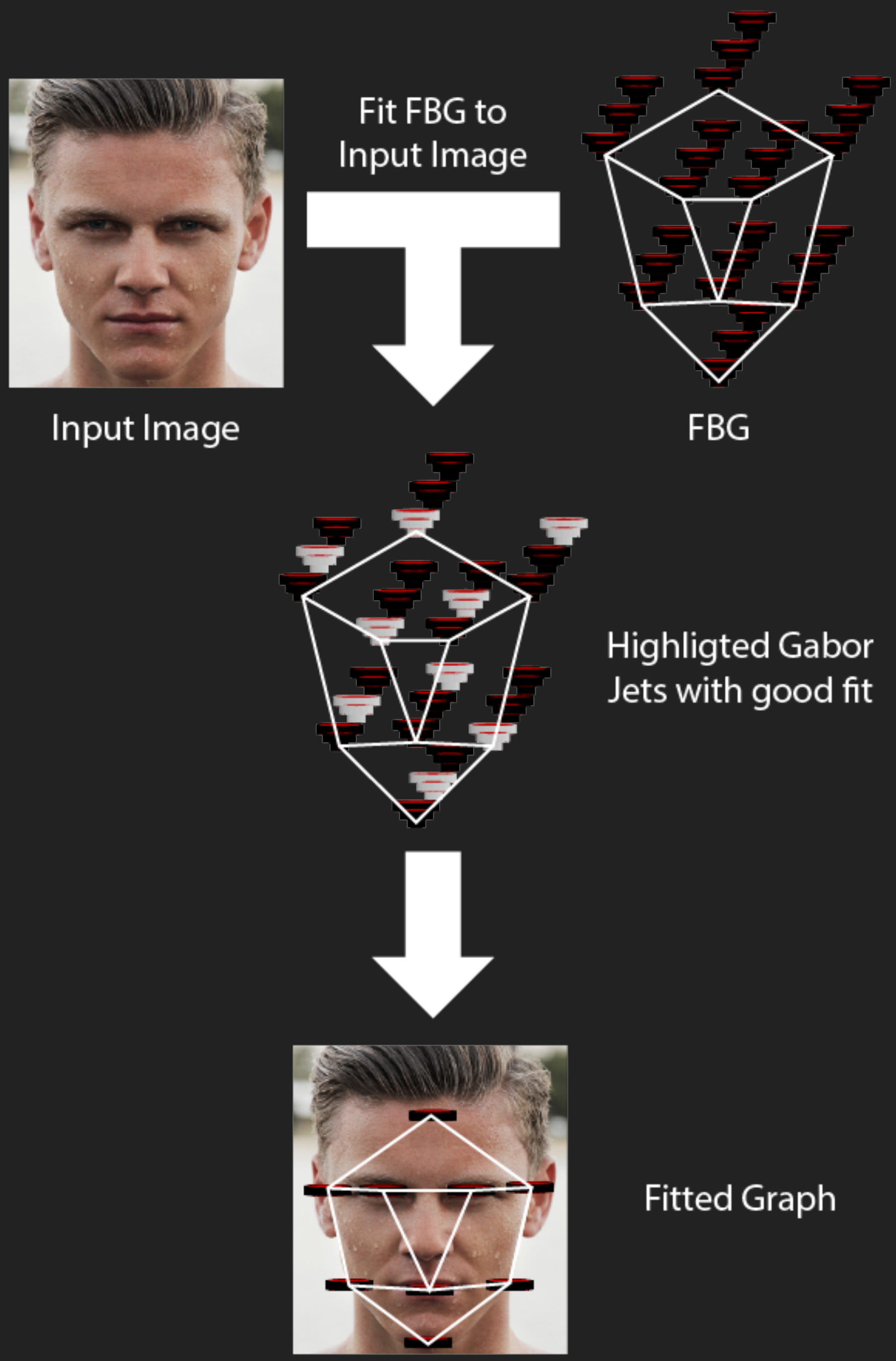


Illustration adapted from *Introduction to Biometrics*, 2011.

REVIEW

- Face Images are typically acquired using a VIS spectrum camera like a camera on your phone
- Viola-Jones uses a cascade classifier to detect faces
- PCA learns a subspace from the training data
- LDA learns a subspace similar to PCA but takes into account the class label (i.e., the subject ID)