Computer Vision

FINAL PROJECT

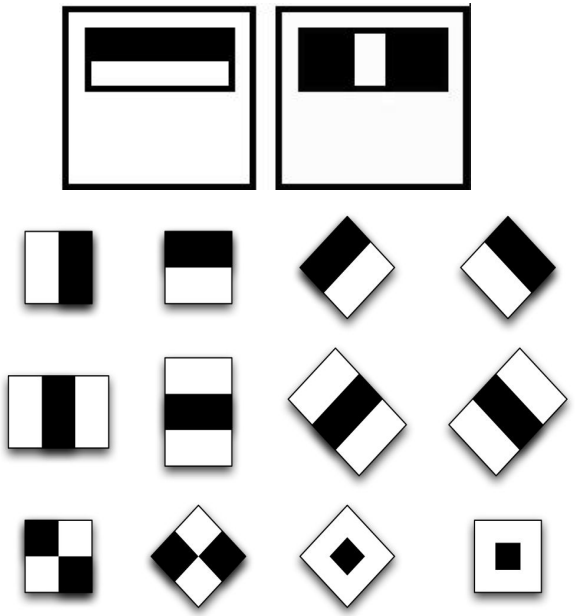
Group#21
Bishoy Samy Bushra
Mohammad Hosni Alzanaty
Mohammad Mohsen Ageez

Agenda:

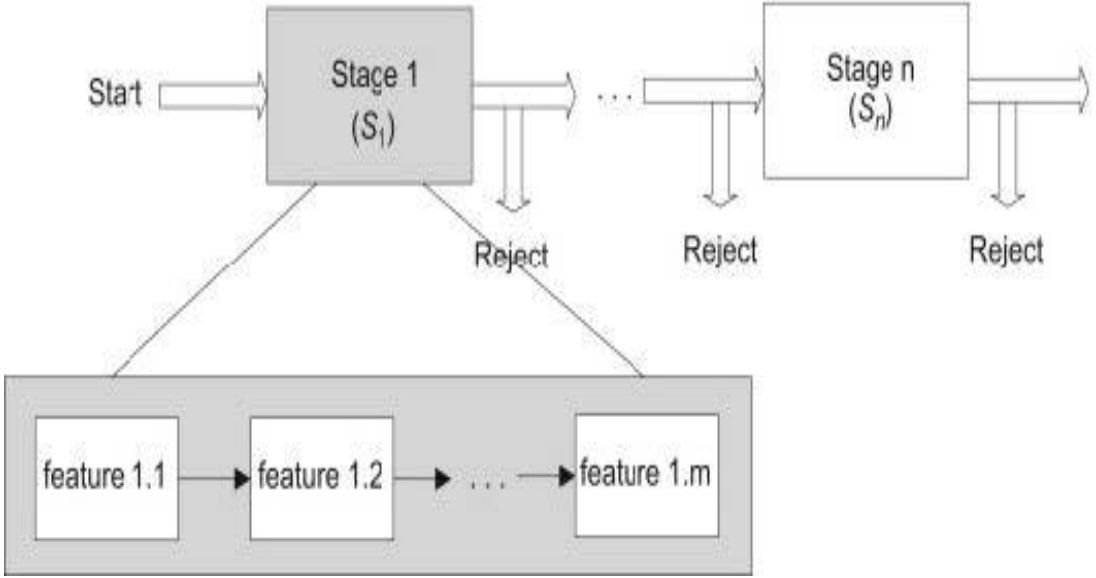
- Face Detection
  - Methodology (Haar-Cascades”Viola-Jones Face Detection”)
  - Implementation (OpenCV-Python)
  - Output
- Face Recognition
  - Methodology
  - Output
  - Performance and ROC curve

Face Detection - Methodology

Viola-Jones algorithm using Haar-Cascade classifiers



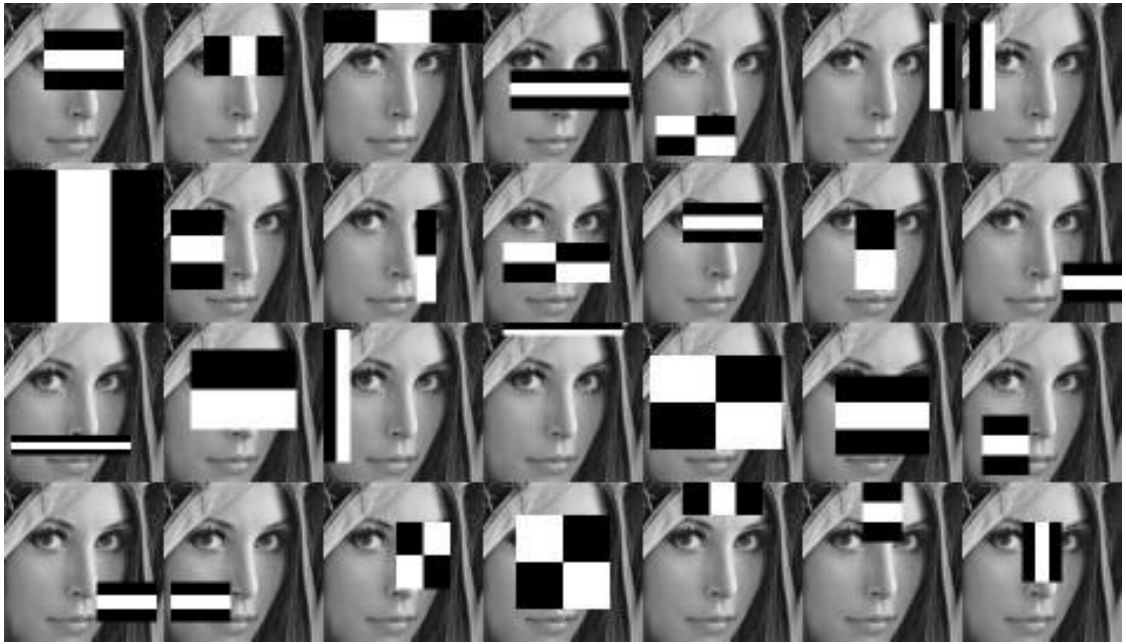
Haar-like features



Cascade of features

Face Detection - Methodology

Viola-Jones algorithm using Haar-Cascade classifiers



(image by Greg Borenstein, shared under a CC BY-NC-SA 2.0 license)

Face Detection - Implementation(*class* CascadeClassifier)

- **CascadeClassifier()**

```
face_cascade = cv2.CascadeClassifier("/home/ageez/haarcascade_frontalface_alt.xml")
```

- **detectMultiScale()**

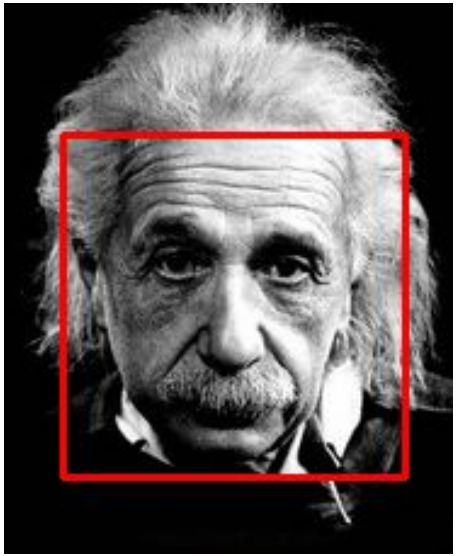
```
faces = face_cascade.detectMultiScale(img, 1.3, 4)
```

- **Drawing the rectangles**

```
for (x,y,w,h) in faces:
    cv2.rectangle(img,(x,y),(x+w,y+h),(0,0,255),2)
cv2.imshow('DetectedFaces',img)
```

Face Detection - Output

Small - colored



Large - grayscale

PCA: Methodology

- Dimensionality Reduction

$$\Sigma = \mathrm{E}\left[(\mathbf{X} - \mathrm{E}[\mathbf{X}])\,(\mathbf{X} - \mathrm{E}[\mathbf{X}])^{\mathrm{T}}\right]$$

$$\mathbf{X}^{\mathrm{T}}\mathbf{X}\boldsymbol{\nu}_i = \boldsymbol{\lambda}_i\boldsymbol{\nu}i$$

$$\mathbf{X}\mathbf{X}^{\mathrm{T}}(\mathbf{X}\boldsymbol{\nu}_i) = \boldsymbol{\lambda}_i(\mathbf{X}\boldsymbol{\nu}_i)$$

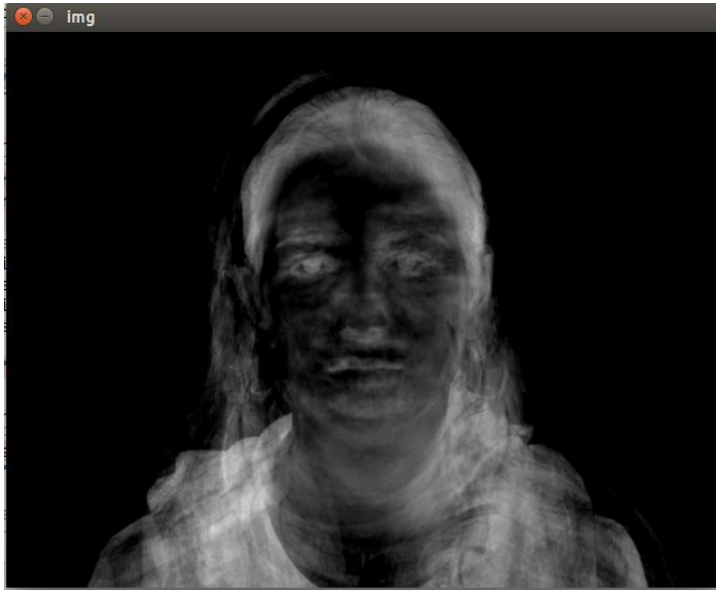
PCA: algorithm

```
def __run(self):
    #   run algorithm
    self.__calculate_cov()

    self.__calculate_eignvectors()

    self.__calculate_eigfaces()

    self.__calculate_kth_coefficient()
```



PCA: Reconstruction

- 10 eig victors.
- 90 %.

$$\mathbf{x} \approx \overline{\mathbf{x}} + a_1\mathbf{v}_1 + a_2\mathbf{v}_2 + \ldots + a_K\mathbf{v}_\mathbf{K}$$

ROC