NUS-ISS
Vision Systems

Module 4 - Foundations of computer vision system (3) - Global feature and representation, part 2

Dr. Tan Jen Hong
Lecturer & Consultant
Institute of System Science
National University of Singapore
issjht@nus.edu.sg

Learning objectives

- Understand the difference between face detection and face recognition
- Understand Viola-Jones algorithms
- Perform face detection
- Perform smile detection

Face detection

- Face detection: locate human faces in an image
- Face recognition: identify human faces in an image
- Face detection answers the question: Where are the faces?
- Face recognition answers the question: Who are the faces?

Source: https://www.kairos.com/blog/face-detection-explained

Face detection

Usage

- Which comes first: detection or recognition
- Energy conservation: smart TV adjusts brightness according to presence of human being in front of it
- Marketers gauge consumer's response when showed marketing materials by detection of face and smile

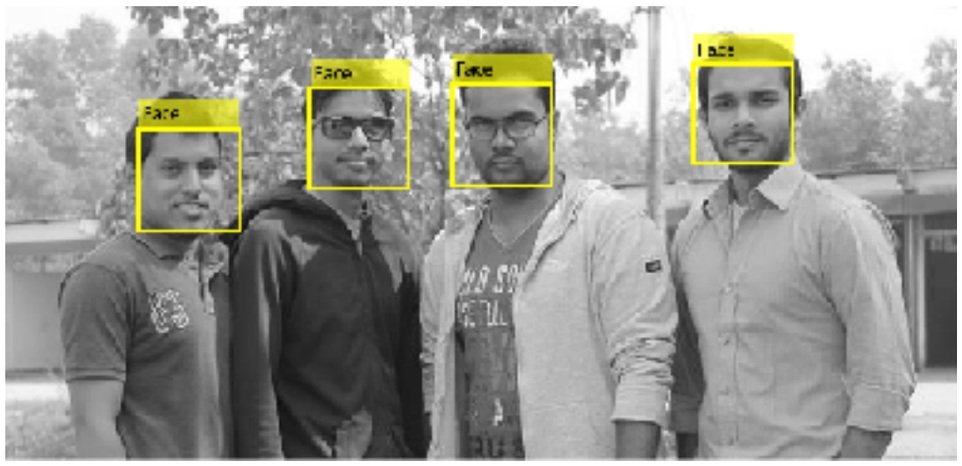
Source: https://www.kairos.com/blog/face-detection-explained

Viola-Jones algorithm

- The first object detection framework to provide competitive object detection rates in ***real-time***, proposed in 2001 by Paul Viola and Michael Jones

- The framework was proposed primarily for face detection, although it can be used for detection of other objects

- Apple used this framework in their public API for many years before they switch to deep neural network few years back



Source: https://www.mathworks.com/matlabcentral/mlc-downloads/downloads/submissions/50077/versions/4/screenshot.jpg

Viola-Jones algorithm

Quick explanation

https://www.youtube.com/watch?v=uEJ71VlUmMQ

Viola-Jones algorithm

Training the algorithm

Understanding and Implementing the Viola-Jones Image Classification Algorithm

Anmol Parande · Follow
Jan 18 · 16 min read

Image classification has been a quickly growing field over the past decade, and the use of Convolutional Neural Networks (CNNs) and other deep learning techniques is growing quickly. However, before CNNs became mainstream, another technique was widely used and continues to be used: Viola-Jones.

Whereas a CNN is a single classifier which looks at a full image and applies matrix operations to arrive at a classification, Viola-Jones takes an ensemble approach. What that means is that Viola-Jones uses many different classifiers, each looking at a different portion of the image. Each individual classifier is weaker (less accurate, produces more false positives, etc) than the final classifier because it is taking in less information. When the results from each classifier are combined, however, they produce a strong classifier.

https://medium.com/datadriveninvestor/understanding-and-implementing-the-viola-jones-image-classification-algorithm-85621f7fe20b

Face detection

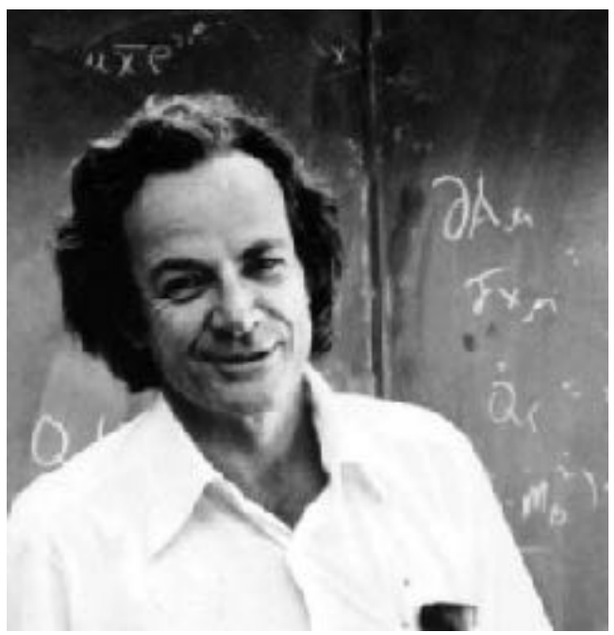
In opencv, using cascade classifier

- Load the image and the necessary libraries

```
> import cv2
> import numpy as np
> import matplotlib.pyplot as plt

> fym = cv2.imread('fym.jpg')
> fymb= cv2.cvtColor(fym,
                     cv2.COLOR_BGR2GRAY)
```

- Image should be converted into gray scale before perform detection



fym.jpg

- Load the xml file

```
> fce = cv2.CascadeClassifier(
        'haarcascade_frontalface_default.xml')
```

Face detection

In opencv, using cascade classifier

- Perform detection

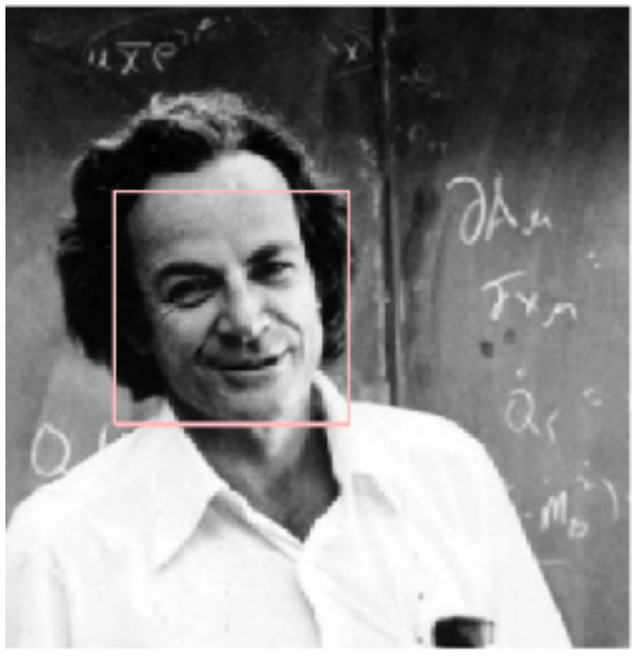
```
> face= fce.detectMultiScale(fymb,
                             scaleFactor=1.3,
                             minNeighbors=5)
```

how much image is reduced at each scale

how many neighbours each candidate rectangle should have to retain it

- `face` is n x 4 array; each row gives the value of x, y, w, h

- Draw the rectangle



dsp

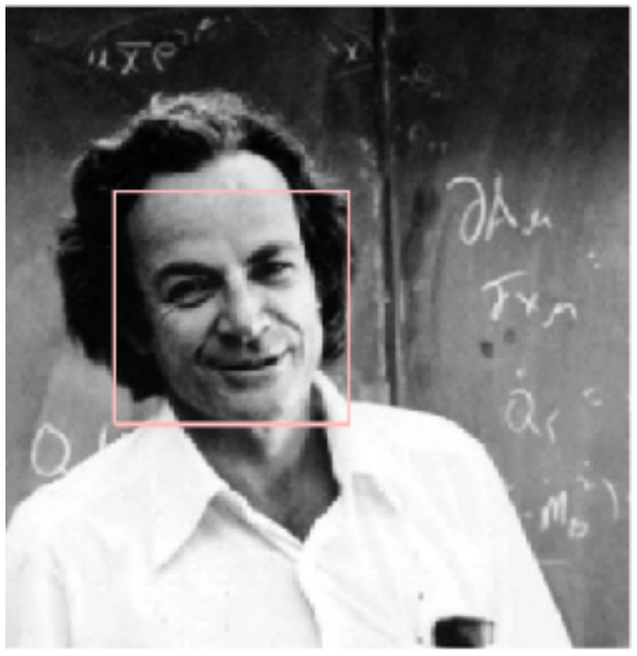
```
> dsp = fym.copy()
> for (x,y,w,h) in face:
    cv2.rectangle(dsp,
                  (x,y),
                  (x+w,y+h),
                  (191,191,255),
                  2)
```

Smile detection

In opencv, using cascade classifier

- Opencv provides trained ‘smile’ and ‘eye’ haar cascade xml files

- Smile detection: search through entire image, or just within the identified face?



Face detection

Previous code

```
> import cv2
> import numpy as np
> import matplotlib.pyplot as plt

> fym = cv2.imread('fym.jpg')
> fymb= cv2.cvtColor(fym, cv2.COLOR_BGR2GRAY)
```

```
> fce = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
> sml = cv2.CascadeClassifier('haarcascade_smile.xml')

> face= fce.detectMultiScale(fymb,
                             scaleFactor=1.3,
                             minNeighbors=5)
> dsp = fym.copy()

> for (x,y,w,h) in face:
      cv2.rectangle(dsp,(x,y),(x+w,y+h),(191,191,255),2)
```

Add this line to load the smile xml file before the for loop

Face detection

Code to detect smiles and draw rectangle

```
.....
> dsp = fym.copy()

> for (x,y,w,h) in face:
    cv2.rectangle(dsp,(x,y),(x+w,y+h),(191,191,255),2)
```

```
fc  = dsp[y:y+h,x:x+w]      ← crop out the face (colour image); rectangle to be drawn on it
fcb = fymb[y:y+h,x:x+w]     ← crop out the face (grayscale image)

smle= sml.detectMultiScale(fcb,                    ← Perform smile detection on the grayscale image
                           scaleFactor=1.3,
                           minNeighbors=10)

for (px,py,pw,ph) in smle:
    cv2.rectangle(fc,(px,py),(px+pw,py+ph),(127,255,127),2)
```

Face detection

Code to detect smiles and draw rectangle

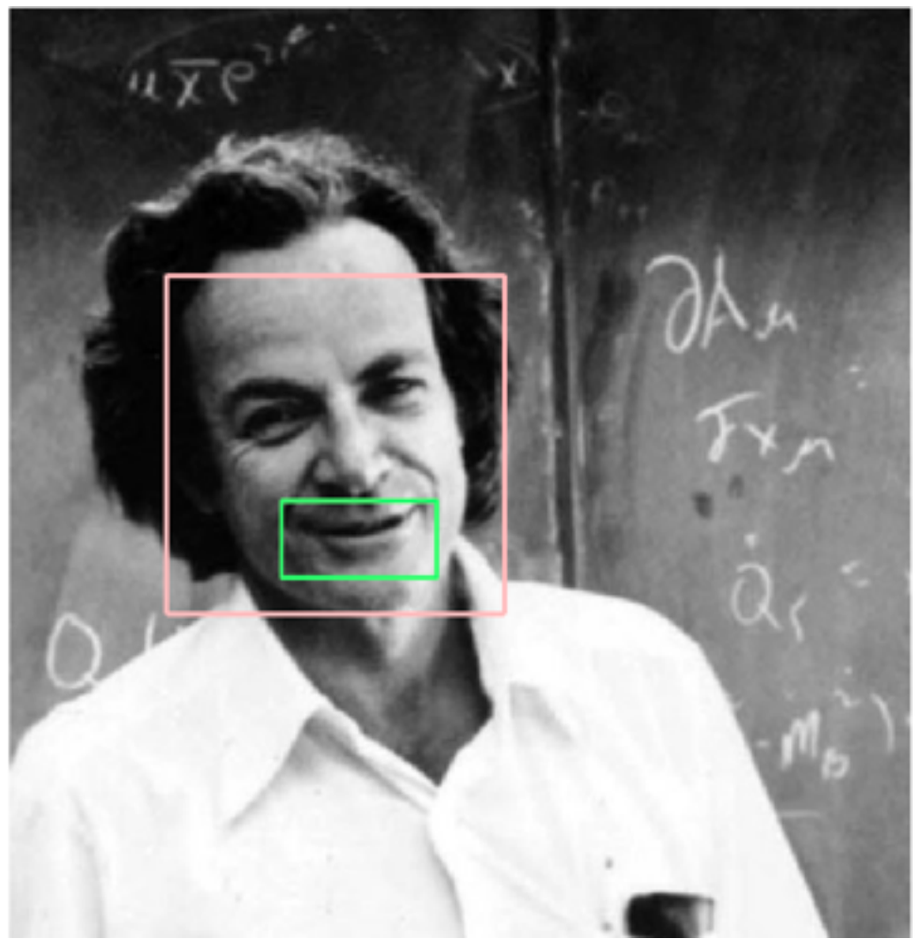
```
.....
> dsp = fym.copy()

> for (x,y,w,h) in face:
    cv2.rectangle(dsp,(x,y),(x+w,y+h),(191,191,255),2)

    fc  = dsp[y:y+h,x:x+w]
    fcb = fymb[y:y+h,x:x+w]

    smle= sml.detectMultiScale(fcb,
                               scaleFactor=1.3,
                               minNeighbors=10)

    for (px,py,pw,ph) in smle:
        cv2.rectangle(fc,(px,py),(px+pw,py+ph),(127,255,127),2)
```



dsp

Viola-Jones algorithm

Advantages

- Extremely fast computation, good for real-time detection
- Position and scale invariant
- Can be applied to other type of object detections

Viola-Jones algorithm

Limitations

- Good and effective only on frontal face
- Not able to cope with faces that rotate around 45 degree
- Sensitive to lighting conditions

https://github.com/opencv/opencv/tree/master/data/haarcascades