Whoo,

A Face Recognition Application based on OpenCV and Android

Shaomin (Samuel) Zhang

Nov 16th, 2015

Outlines

- A Brief Introduction of Face Recognition
- Motivations
- OpenCV & Android
- The Algorithms
- Software Development: Design and Implementation
- Demonstration
- Conclusion and Future Work

Face Recognition & Face Detection

Face Detection

It is to find the face(s) in the given image. It can tell where the faces are in an image but can't tell who they are.

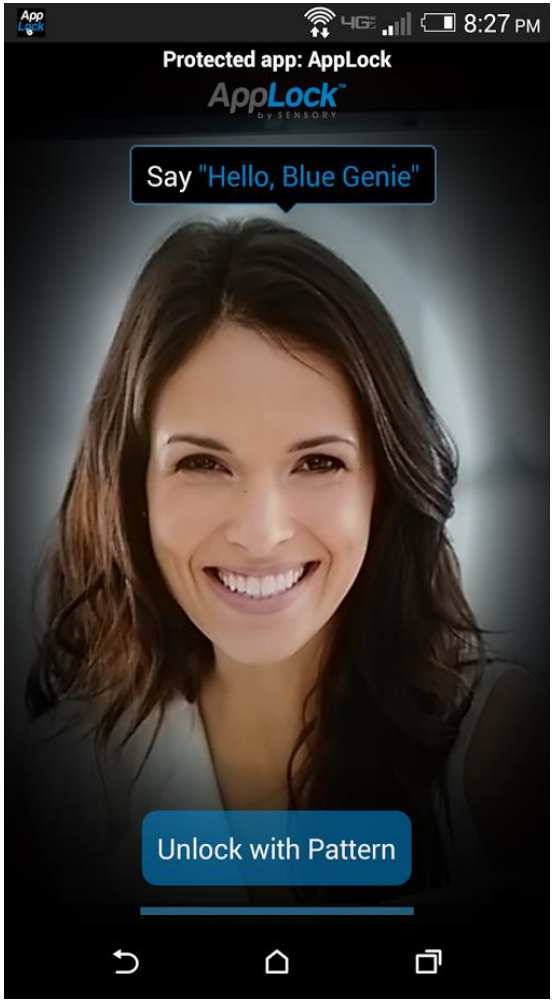
Face Recognition - Verification

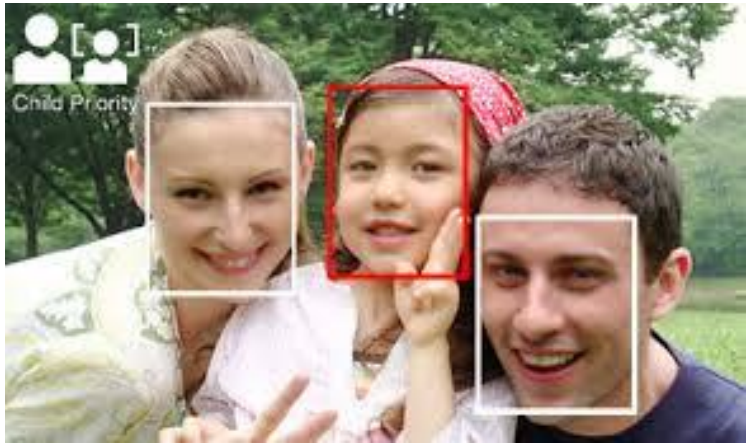
It compares the given face image against a template face whose identity has already been claimed, and gives a yes or no decision. (It is a one-to-one match)

Face Recognition – Identification

It is a one-to-many matching process that compares a query face image against all the template images in a face dataset to determine the identity of the query face.

Face Recognition & Face Detection





Face Recognition (1/3)

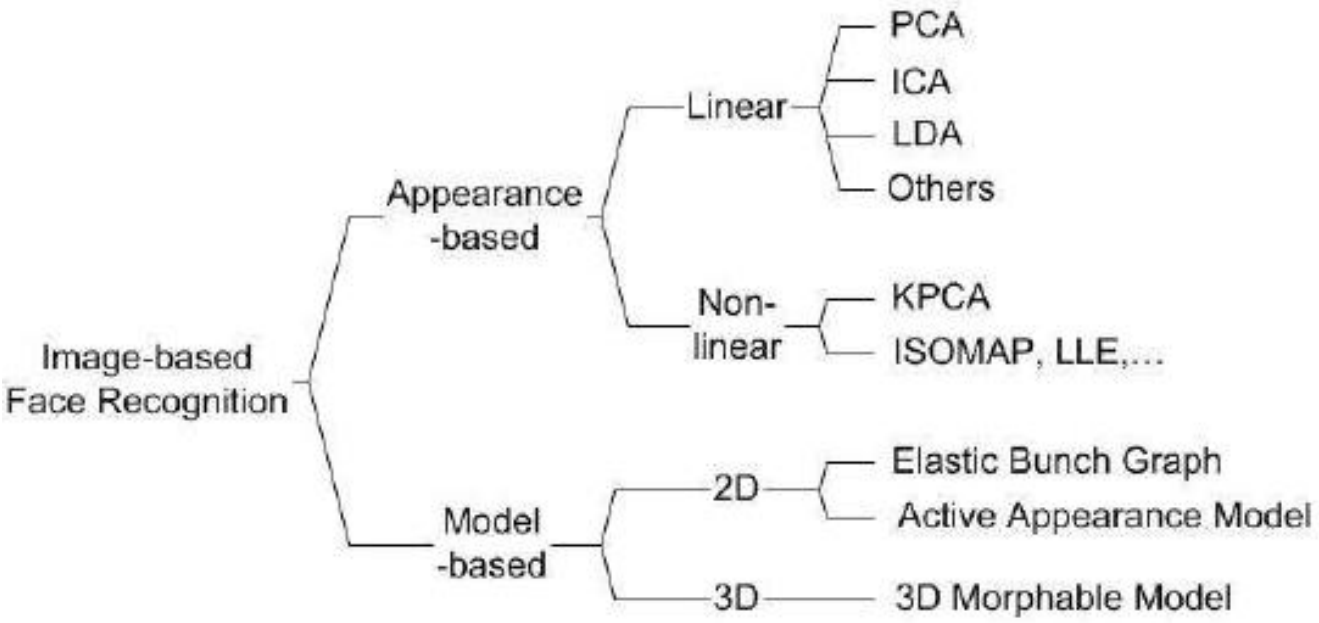
Feature-based Methods (from 1960s):

Some semi-automated system was used for facial recognition to locate the features(such as eyes, ears, nose and mouth) on the images, and multiple specific subjective markers such as hair color and lip thickness to automate the recognition.

View-based Methods (Eigenfaces, 1991):

The real milestone achievement of face recognition was the publication of the *Eigenfaces* method by Turk and Pentland. A face image is a 2-D array of pixel intensity values or a high-dimensional vector, and some template matching methods are used to statistically analyze the distribution of face images and derive its low-dimensional representation.

Face Recognition (2/3)



Lu, X. “Image Analysis for Face Recognition”. Department of Computer Science and Engineering Michigan State University, 2003.

Face Recognition (3/3): Applications

The face recognition based security system of Boston's Logan International Airport had a 61.4% success rate before it was scrapped in 2003.

Apple published a face recognition patent early in 2012, and later deployed into its iPhone to let user unlock the device using his or her face. Meanwhile, Samsung deployed the same feature on its Galaxy S series products.

Motivations

Face Recognition demonstration in Microsoft's On-campus Tech Festival, Xi'an China, 2003.

Face recognition research achievements are getting more and more mature and it seems technically feasible to build practical software products based on existing open-source project like OpenCV.

FaceL, Colorado State University, 2009

LuxandFace SDK, 2010-2015

Face++, 2015

Microsoft's How-Old.net, 2015

…

OpenCV

OpenCV (Open Source Computer Vision Library) declares to support several hundred computer vision algorithms.

It published an additional module which includes 3 face recognition algorithms from version 2.4.9 .

It even defined a set of standard APIs for face recognition:
http://docs.opencv.org/2.4.6/modules/contrib/doc/facerec/facerec_api.html

It is cross-platform, lightweight, and efficient. It supports Android! (OpenCV4Android)

Android

The dominant operating system without suspense in this so-called mobile internet era.

It is open-source and has a globally large and active community of developers and enthusiasts.

The Android framework provides powerful but friendly support for cameras and camera features that are available on devices, allowing us to capture images and videos in our own applications.

The Algorithms supported by OpenCV

Eigenfaces suggested a basic idea: a face image is a point of a high-dimensional image space, and if we can find its lower-dimensional representation through which classification becomes possible, the face recognition will be resolvable.

Fisherfaces can be considered as an improvement of *Eigenfaces* to a certain extent.

LBPH (Local Binary Patterns Histograms) is a classic feature-based algorithm that works much better in poorly illuminated environments, and in the scenarios of our software.

Eigenfaces & PCA

The training set $X = \{ x_1, x_2, \ldots, x_n \}$,

The mean face $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$

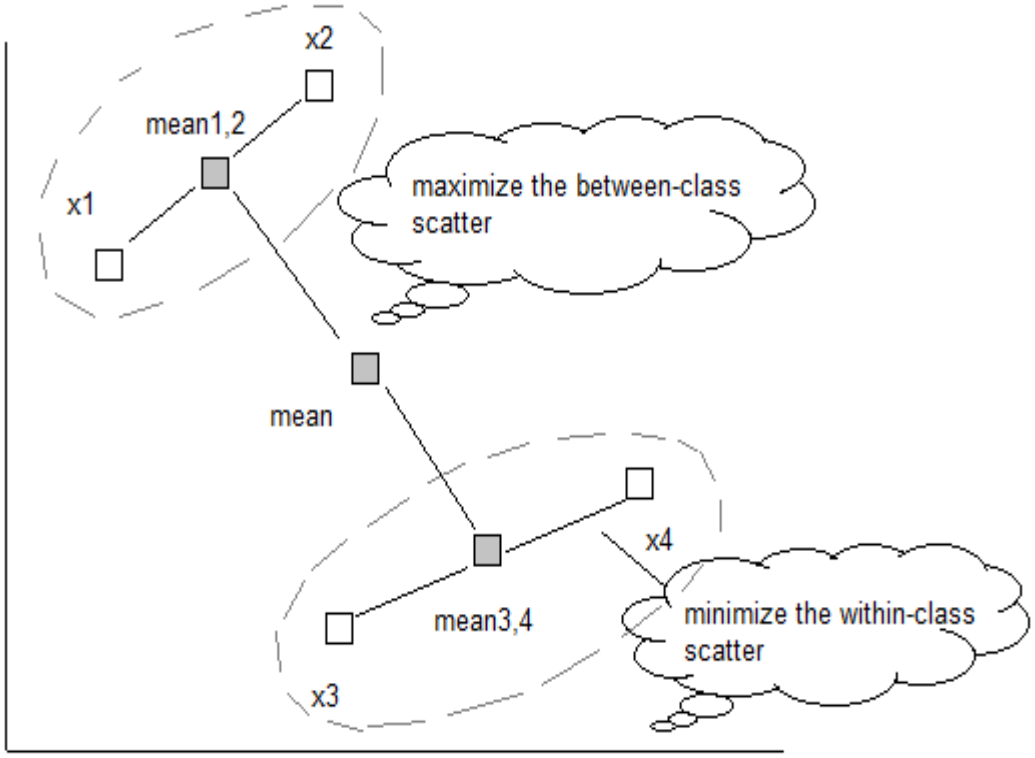
$y_i = \mathrm{W}^{\mathrm{T}} (x_{\mathrm{i}} - \mu)$

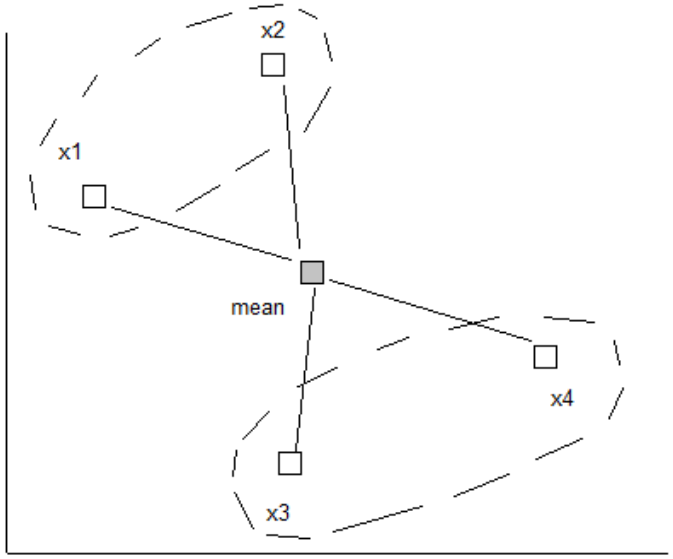
$x_i = \mathrm{W}^{\mathrm{T}} y_{\mathrm{i}} + \mu$

The projection feature vector space $Y = \{ y_1, y_2, \ldots, y_n \}$

Fisherfaces & LDA

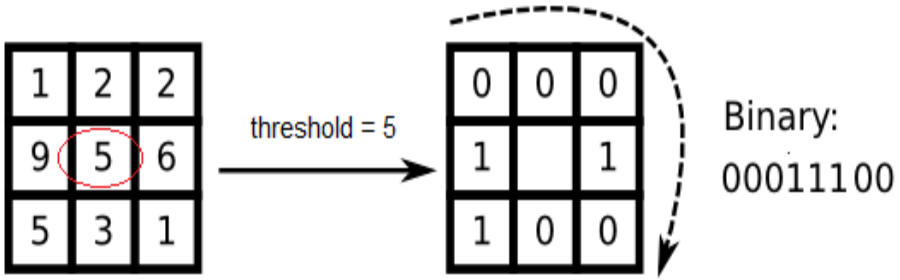
LDA distinguishes Subsets, and maximizes the Between-class Scatter while minimizes the Within-class Scatter



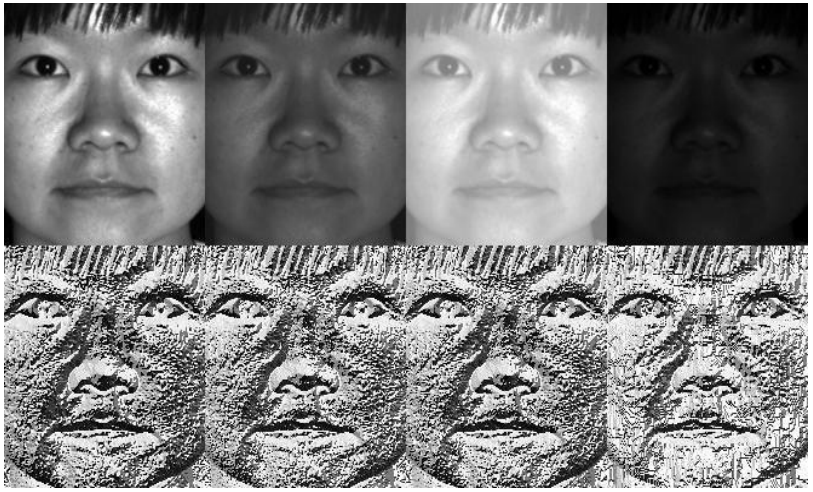


PCA does not consider any Classes information when calculating the Covariance Matrix and the Principal Components

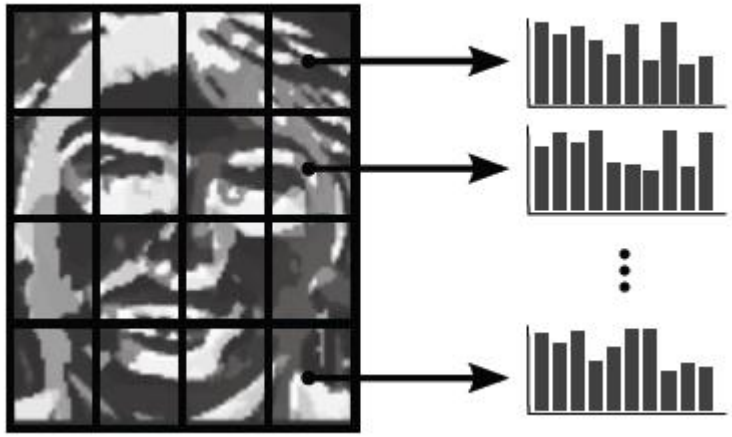
LBPH (1/2)

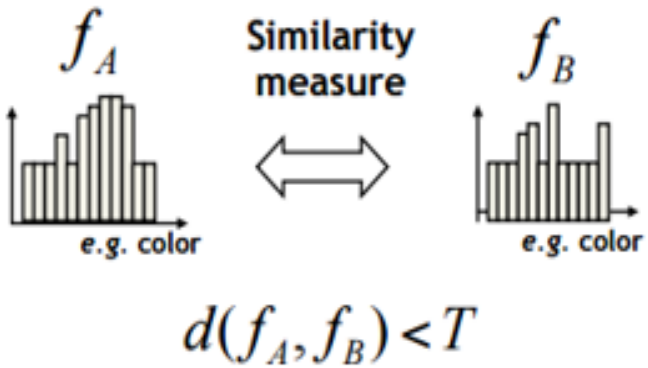


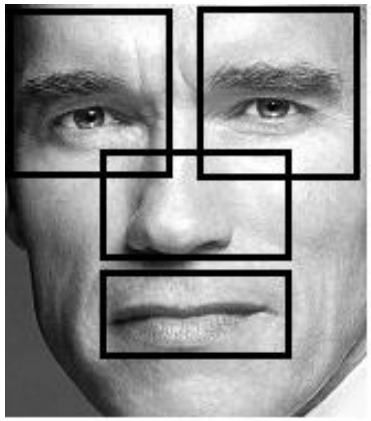
LPBH method contributes an unexpected advantage for face recognition because the processed image does not suffer from illumination variations at all !!

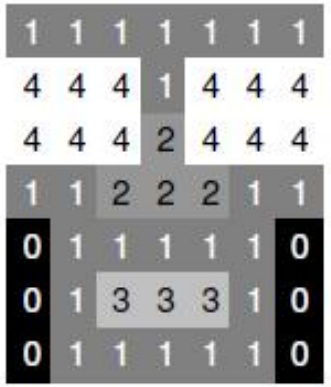


LBPH (2/2)




f_A
Similarity measure
f_B
e.g. color
e.g. color
$d(f_A, f_B) < T$




1 1 1 1 1 1 1
4 4 4 1 4 4 4
4 4 4 2 4 4 4
1 1 2 2 2 1 1
0 1 1 1 1 1 0
0 1 3 3 3 1 0
0 1 1 1 1 1 0

Specification of Whoo

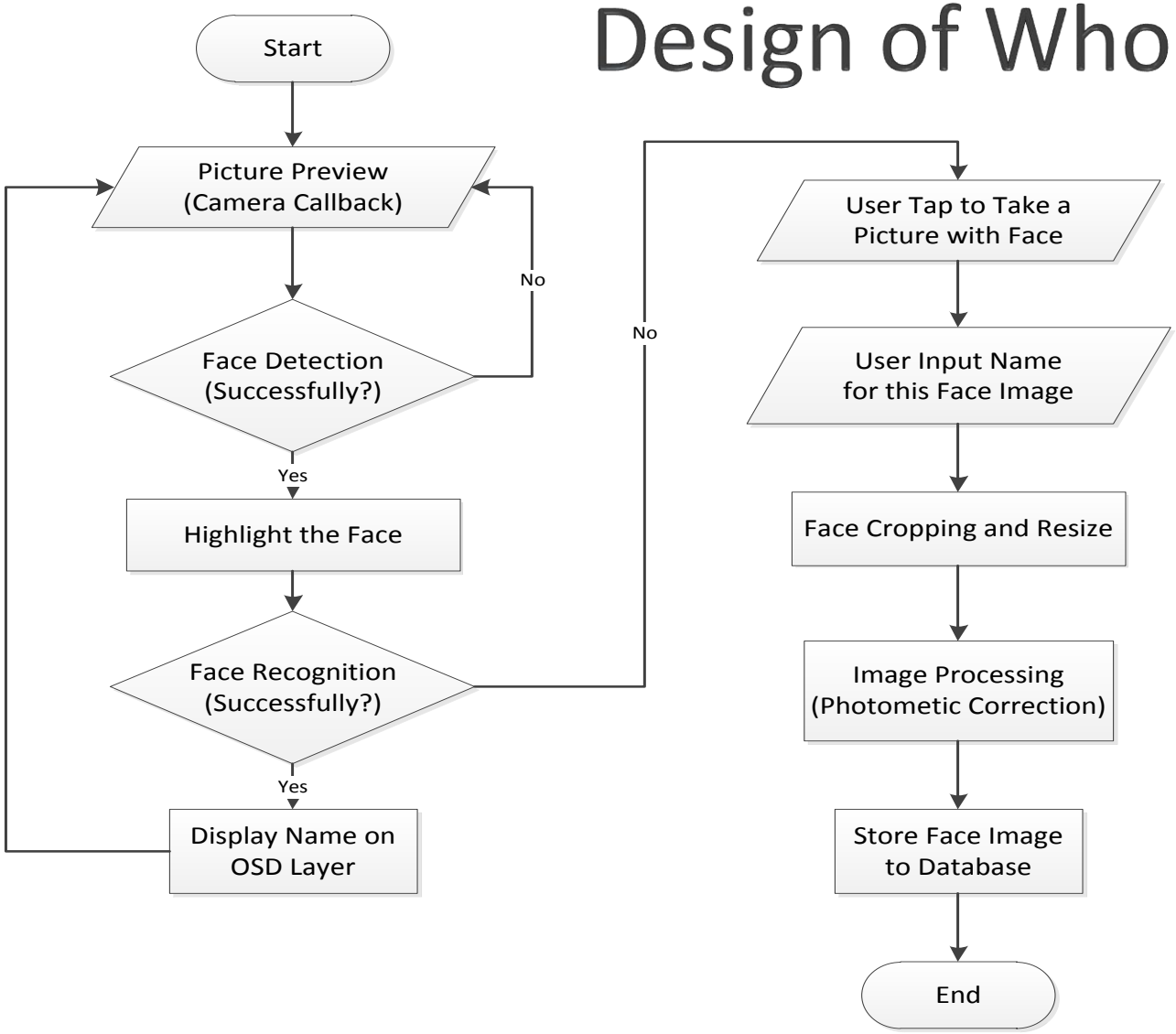
- **Basic Edition**

  A stand-alone Android client application that takes pictures, captures faces, and executes face recognition using the small-range dataset on its local storage.

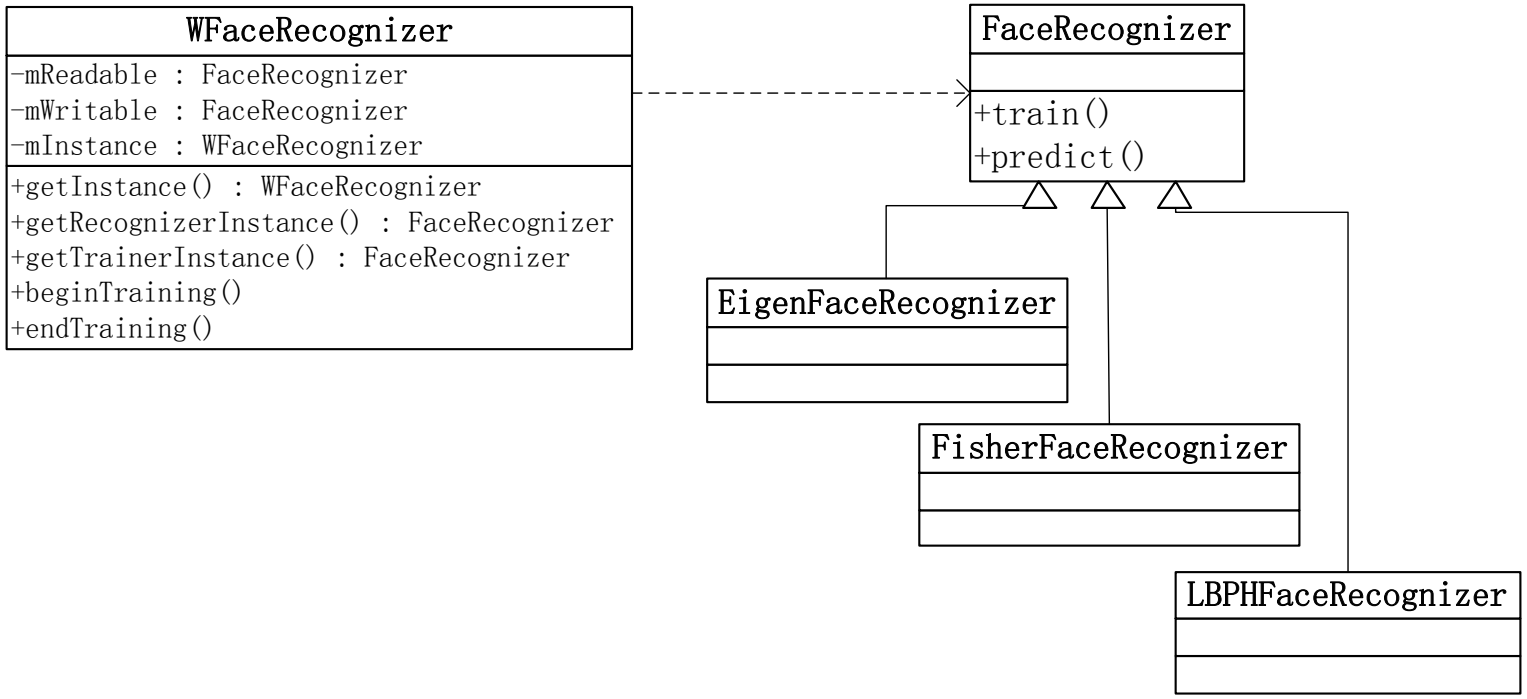
- **Enhanced Edition**

  Compatible with all features of the basic edition but transmits face data to a remote server that maintains a large-scale dataset and runs the same algorithms on the server side.

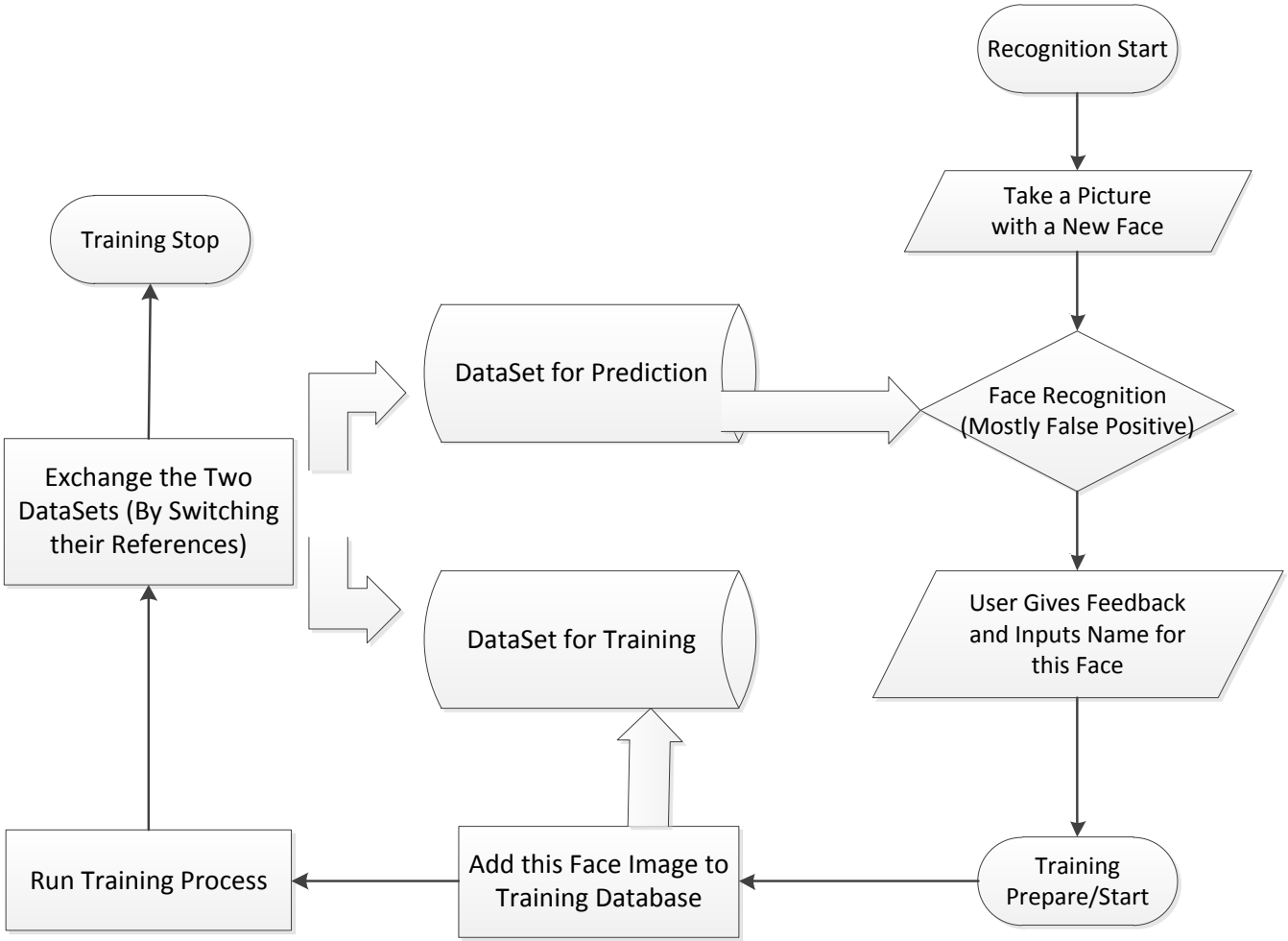
Design of Whoo (1/4)



Design of Whoo (2/4)



Design of Whoo (3/4)



Design of Whoo (4/4)

Camera-based GUI

- ✓ Make full use of the camera preview feature
- ✓ Make the user interactions simple and easy-to-do

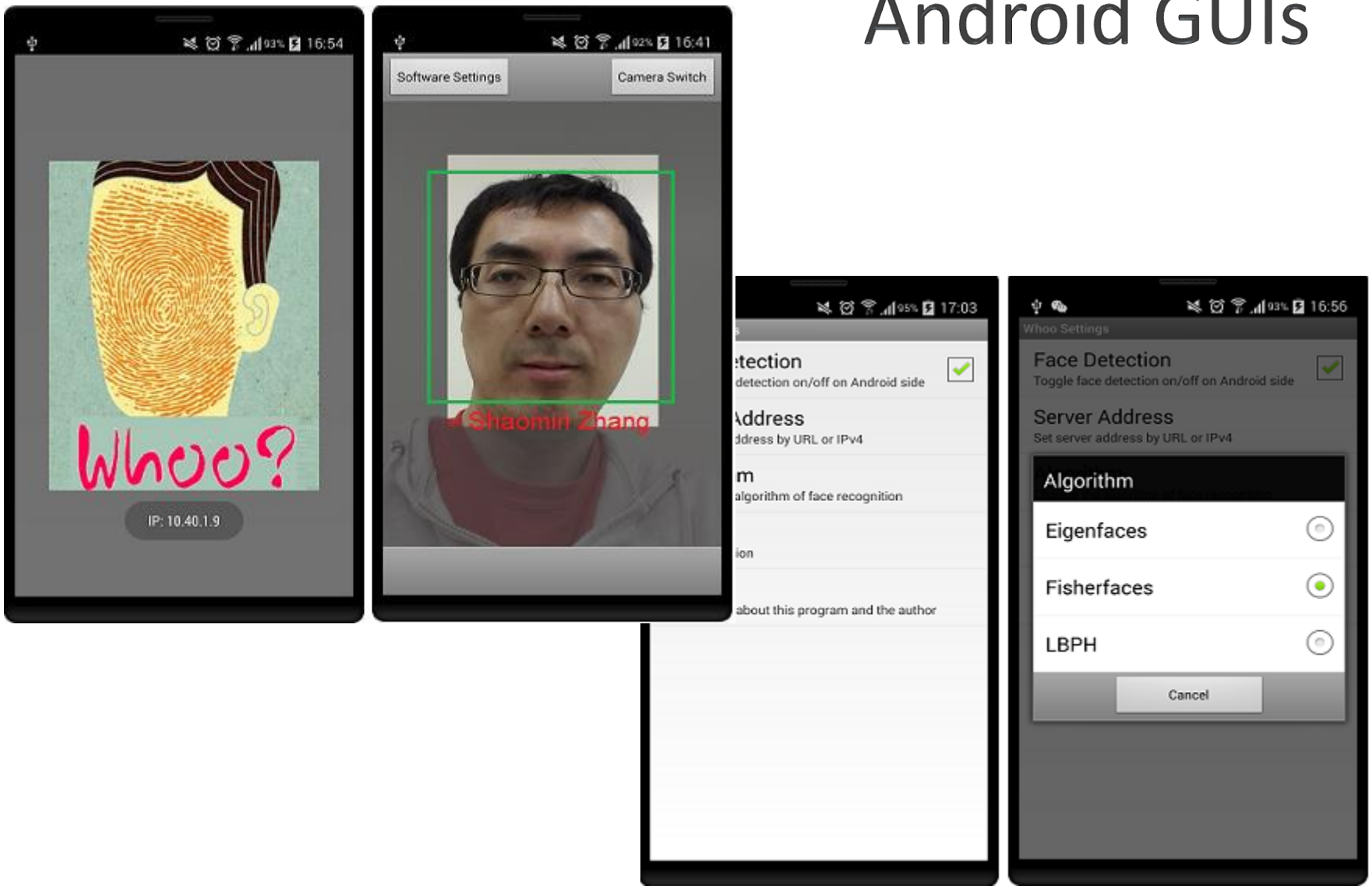
Multi-threading

- ✓ Let the two procedures ***training*** and ***prediction*** be able to run simultaneously

Data Management

- ✓ Duplicate datasets and duplicate cv:FaceRecognizer instances
- ✓ Name input history kept in local file

Android GUIs



Implementation

Development/Runtime Environment

- ✓ Android Studio 1.0.1 (build on Dec 2014)
- ✓ Android SDK 4.4.2 (API 19)
- ✓ OpenCV 2.4.9
- ✓ Samsung Galaxy S4 + Android 4.4.2 (Kitkat)

Project and Source Code

- ✓ 23 java source files
- ✓ 1 native C++ source file
- ✓ 3500 lines of source code

Test Results and Observations

We collected 120 face images of 44 persons for testing.

Eigenfaces needs about 10 different images for each person so that the algorithm can work well and give acceptable results.

Fisherfaces works good when there are not many people but each person has many images in the dataset.

LBPH works good enough in the scenario of our software, most people can be correctly identified by given only 1 or 2 images.

One loop of training costs about 5s, 8s and 2s respectively.

Future Work

- Let the face recognition datasets connected to the phone contact list and other 3rd-party social media software.
- Add features to scan business cards and detect the text information, so that it can help build one's personal contact list by scanning business cards and faces.
- Deploy the face recognition from mobile phones to wearable devices, like Google Glasses.
- Continue the development at server side and provide face recognition web services by servers along with OpenCV and CUDA installed.
- Move the face recognition web services onto AWS or other could platforms.
- Continue to improve the performance and success-rate of current algorithms and import better algorithms.

References

- OpenCV Official Website
  http://www.opencv.org
- Face Recognition Homepage
  http://www.face-rec.org
- 16 Face Recognition related Papers
- COSC 5340-01 Computer Vision by Dr. Zhang
- OpenCV and three other open-source projects
- Google, Wikipedia, StackOverflow.com

Thank you!