Technische Universität Kaiserslautern

Institute of
Integrated Sensor Systems
Department of Electrical and Computer Engineering

Hyena
Acquisition, Processing, Training and Testing process of images using python and openCV

Francisco Javier García Rosas

Marzo 2015

Prof. Dr.-Ing. Andreas König

General description of the project

- This project develops image **data acquisition GUI in Python**. The GUI also include the camera interface driver.
- The software is expected to be **compatible with standard USB** Web-camera and Video capture card.
- The GUI provides **Live-view capturing feature** as well as data manipulation.

Objectives:

* Develop an *interface* to drive accessing to connected camera.
* Develop a *live-view capturing feature*.
* Develop an *Image and data manipulation* function.
* Implement an *example of recognition task* from captured image by using develop software.

Overview

1. Presentation of case study
2. Presentation of Hyena App
3. Acquisition module
4. Processing module
5. Training module
6. Online test module
7. QuickCog interface and example
8. Conclusions
9. Future work

1. Presentation of case study

Develop a software application for Real-time Identification of people's faces based on previous images, which were taken from a public dataset of images in internet and also photos which were taken to some people in the university of Kaiserslautern.

Characteristics:

- Between 10 ~ 50 photos per person
- Relative small size of images (200x 200 ~ 640x480)
- Between 9 ~ 50 individuals (memory restrictions for training algorithms)
- Learning algorithm focused on detection and recognition of faces, which are already implemented in OpenCV
- Two image's dataset: GoodSamples and BadSamples for comparisons and analysis

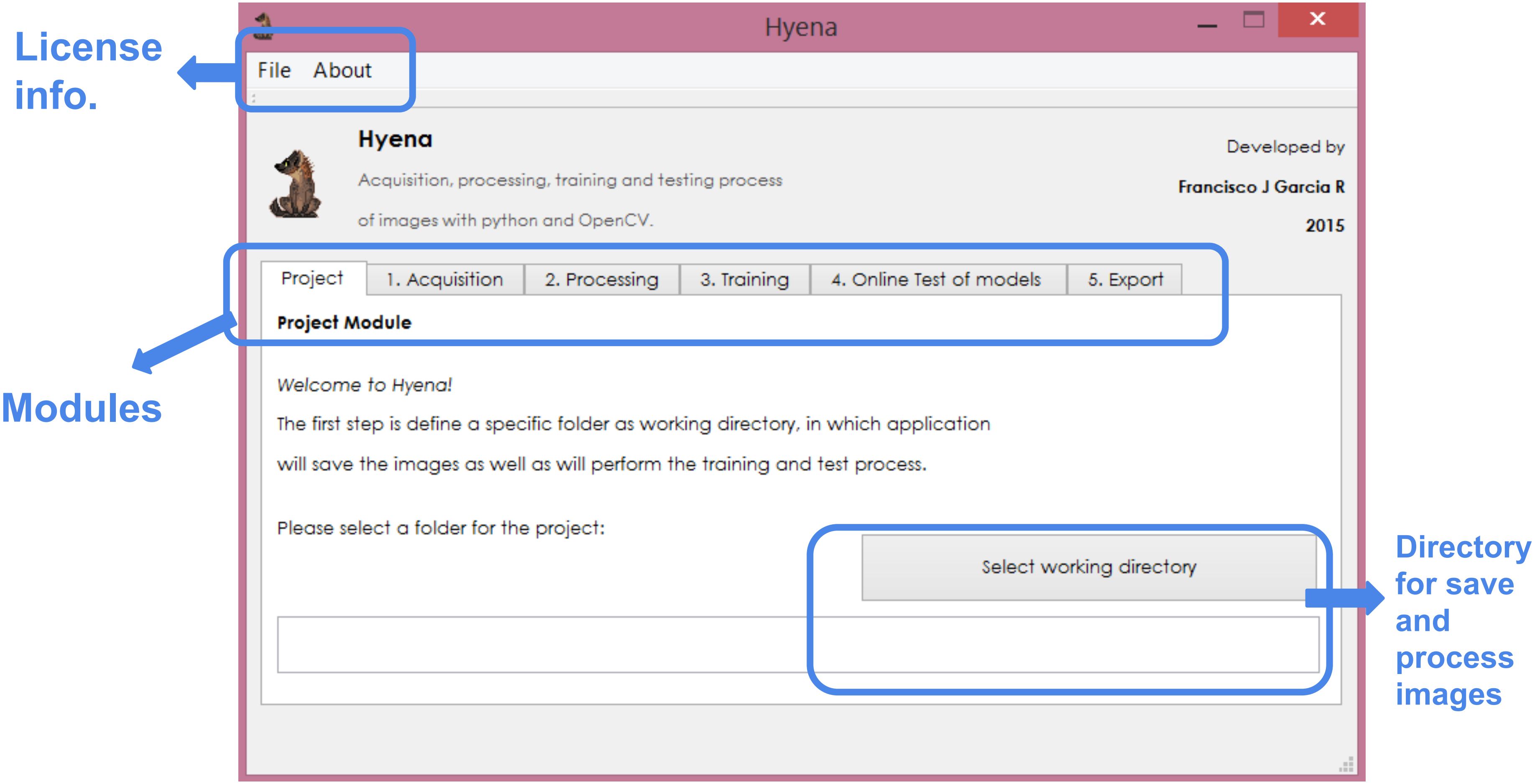
2. Presentation of Hyena

Hyena is a software application developed with Python, which can manage the process of acquisition, processing, training and online testing of images using OpenCV and SciPy algorithms.

Main Characteristics:

- **Multi-platform** and **open source**
- Acquisition module:
  - Take images from camera or select existing images
- Processing module:
  - Apply **filters, edges detectors, change of shape**, etc…
- Training module:
  - Focus on **detection and recognition of faces**
- Test online module
  - **Real-time application for identification** of people's faces
- Export dataset to QuickCog

2. Presentation of Hyena



Hyena

Acquisition, processing, training and testing process

of images with python and OpenCV.

Developed by

Francisco J Garcia R

2015

Project Module

Welcome to Hyena!

The first step is define a specific folder as working directory, in which application

will save the images as well as will perform the training and test process.

Please select a folder for the project:

3. Acquisition Module

Module’s Characteristics

The application uses a OpenCV interface for acquisition of images, which is widely supported it and also can manage different types of commercial cameras.

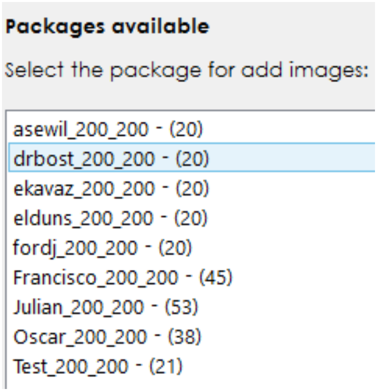
The module also allows to select existing images from external devices such as external hard disks or pen drives.

The user can create, add images and delete packages.

Each **package** represents a **Class**

A **package is a folder**, in which will be save and process the images.

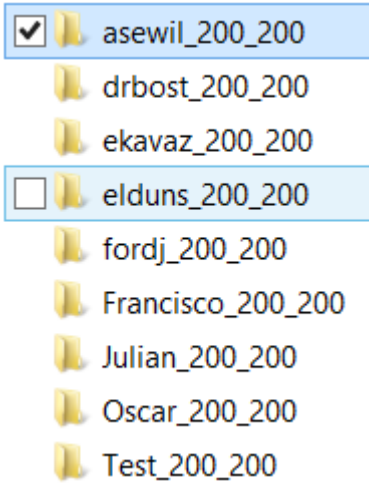
Study case



Packages available

Select the package for add images:

asewil_200_200 - (20)
drbost_200_200 - (20)
ekavaz_200_200 - (20)
elduns_200_200 - (20)
fordj_200_200 - (20)
Francisco_200_200 - (45)
Julian_200_200 - (53)
Oscar_200_200 - (38)
Test_200_200 - (21)

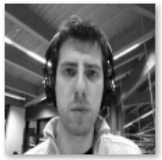


☑ asewil_200_200
drbost_200_200
ekavaz_200_200
□ elduns_200_200
fordj_200_200
Francisco_200_200
Julian_200_200
Oscar_200_200
Test_200_200

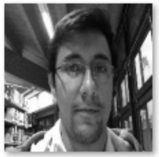
For this project was selected **9 individuals,** each of them has between **20 ~ 53 images**

fb865e7a.jpg

ff293ad3.jpg



3c2a4a4d.jpg



1ed4b2ea.jpg

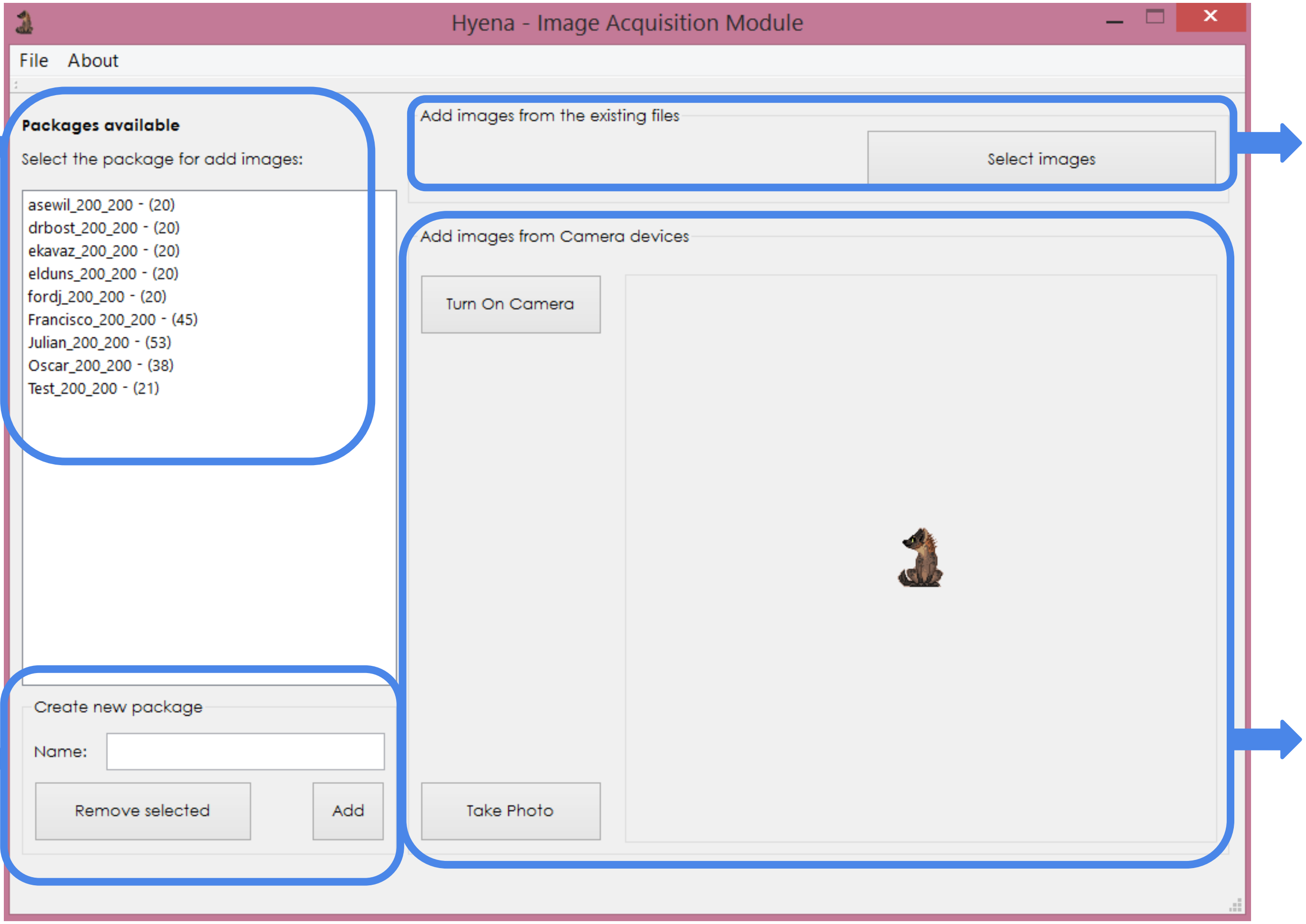
3. Acquisition Module

Packages included in the project

Name of each class and number of images.

Manage packages:

Create and remove



Packages available

Select the package for add images:

- asewil_200_200 - (20)
- drbost_200_200 - (20)
- ekavaz_200_200 - (20)
- elduns_200_200 - (20)
- fordj_200_200 - (20)
- Francisco_200_200 - (45)
- Julian_200_200 - (53)
- Oscar_200_200 - (38)
- Test_200_200 - (21)

Create new package

Name:

Add images from the existing files

Add images from Camera devices

External image files

Camera interface

4. Processing Module

Module's Characteristics

Options available:

- Color shape
  - Resize, laplacian, smooth, sobel, erosion, dilation etc
- Thresholding
  - Binary, Truncate, to zero, adaptative and Otsu.
- Edge detector
  - Canny edges algorithm

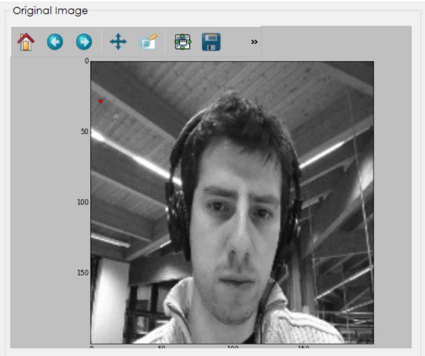
Functions:

Show a preview image with the filters selected.

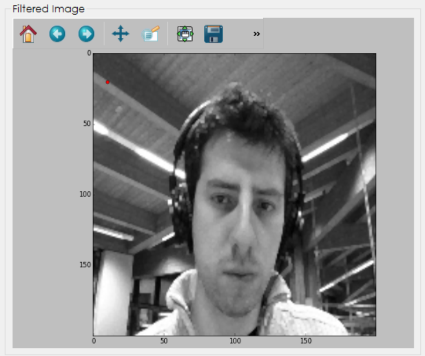
Batch processing for applying the filters selected to the images in each package

Study case

Resize all images to 200x200 for normalize data and reduce computational effort.



RGB to gray-scale for improving the accuracy of the learning algorithms (avoid misleading with colors)



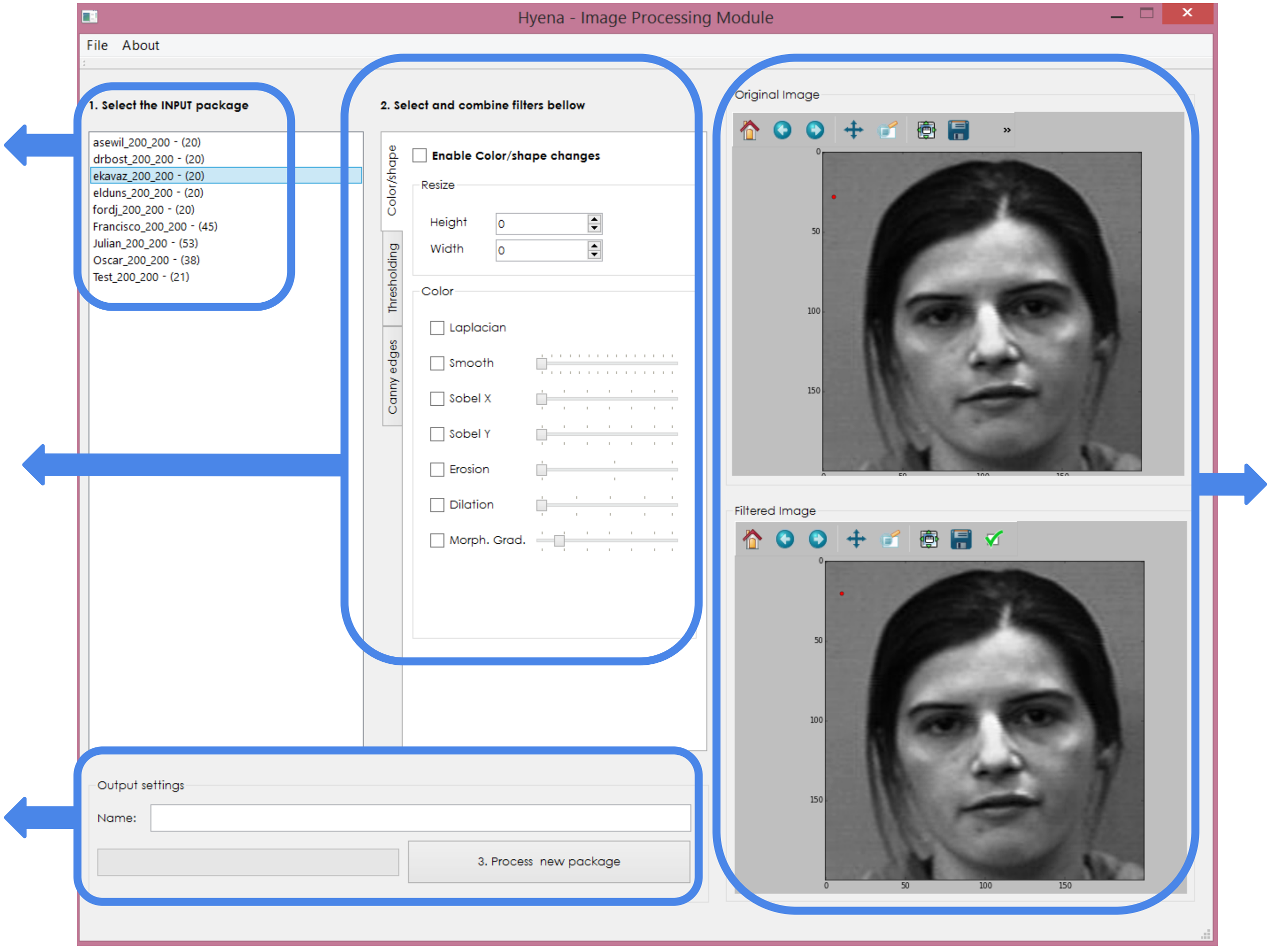
Slightly **dilated and smoothed** - focused on shape of face.

4. Processing Module

Packages included in the project

Filters available

Output configuration



Hyena - Image Processing Module

File About

1. Select the INPUT package

- asewil_200_200 - (20)
- drbost_200_200 - (20)
- ekavaz_200_200 - (20)
- elduns_200_200 - (20)
- fordj_200_200 - (20)
- Francisco_200_200 - (45)
- Julian_200_200 - (53)
- Oscar_200_200 - (38)
- Test_200_200 - (21)

2. Select and combine filters bellow

Color/shape | Thresholding | Canny edges

□ Enable Color/shape changes

Resize

Height 0

Width 0

Color

□ Laplacian

□ Smooth

□ Sobel X

□ Sobel Y

□ Erosion

□ Dilation

□ Morph. Grad.

Output settings

Name:

3. Process new package

Original Image

Filtered Image

Preview images:

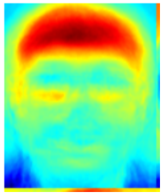
Original and filtered

5. Training module

Module’s Characteristics

Algorithms which were included:

Eigenfaces:



A high-dimensional dataset into few meaningful dimensions account for most of the information: PCA

Fisherfaces:

Linear Discriminant Analysis performs a class-specific dimensionality reduction

Local Binary Patterns Histograms:

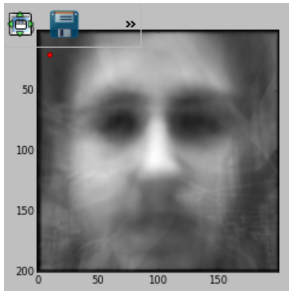
Summarize the local structure in an image by comparing each pixel with its neighborhood.

OpenCV docs: http://docs.opencv.org/modules/contrib/doc/facerec/facerec_tutorial.html

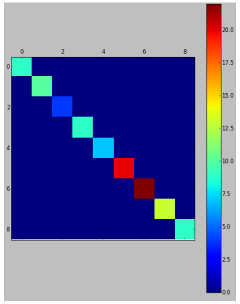
Study case

Good samples: face delimited automatically and preprocessed

Mean face



Confusion matrix

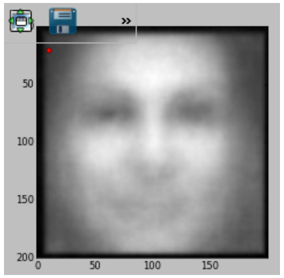


Example

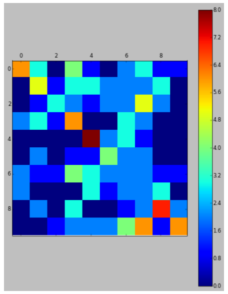
ed0c3c15.jpg

Bad samples: original images with noise

Mean face



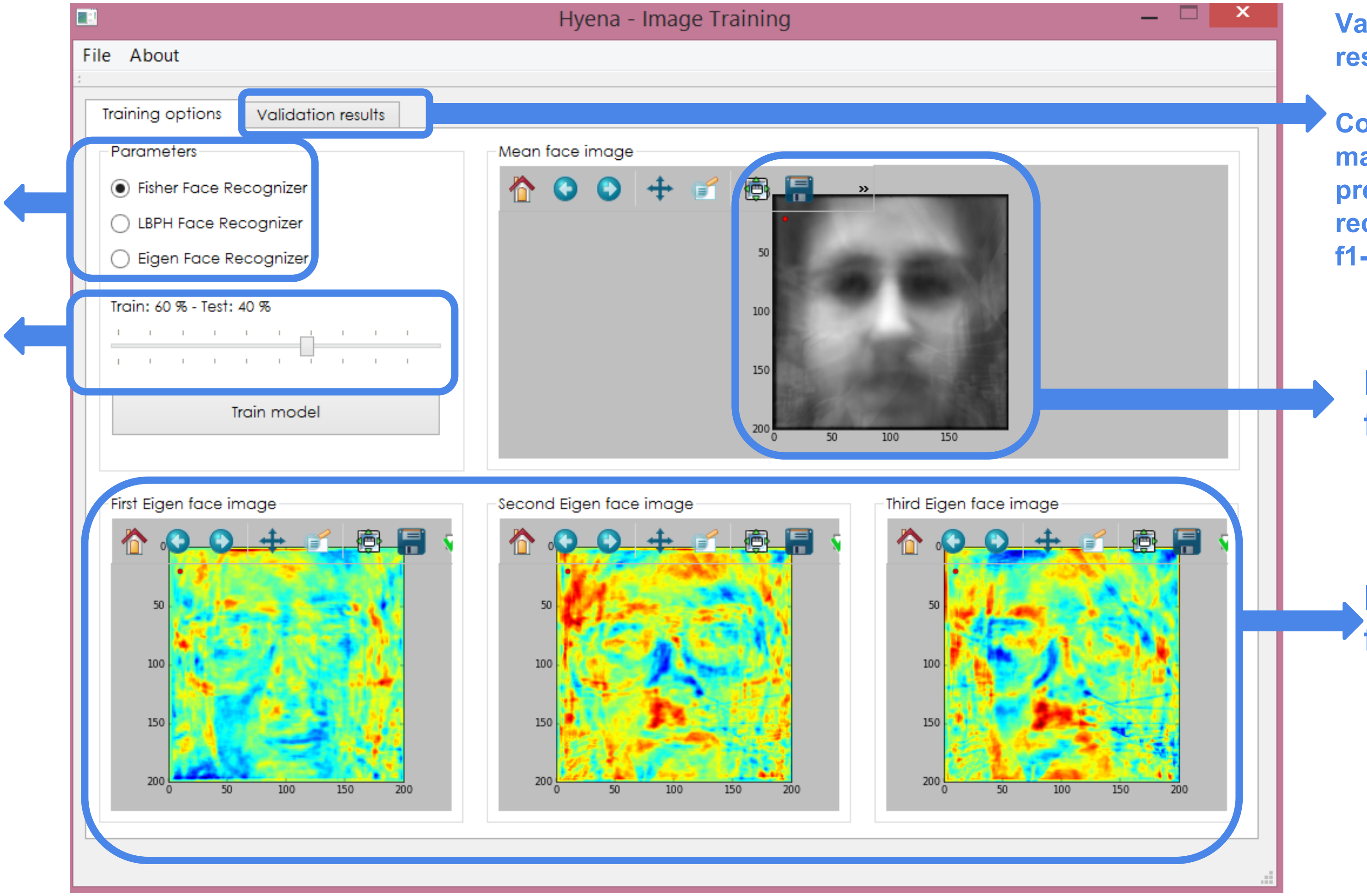
Confusion matrix



Example

118.jpg

5. Training module



Hyena - Image Training

File About

Training options | Validation results

Parameters

● Fisher Face Recognizer
○ LBPH Face Recognizer
○ Eigen Face Recognizer

Train: 60 % - Test: 40 %

Train model

Mean face image

First Eigen face image

Second Eigen face image

Third Eigen face image

Select algorithm

Define size of training and testing datasets

Validation results:

Confusion matrix, precision, recall and f1-score

Mean face

Eigen faces

6. Online test module

Module's Characteristics

This module performs real-time recognitions tasks, based on models which were previously created in the training module.

1. Preprocessing of image: resize to 200x200 and convert to grayscale.
2. Find regions of faces in the image with Cascade Classification algorithms.

   http://docs.opencv.org/modules/objdetect/doc/cascade_classification.html
3. Cuts the region and detect individuals based on models
4. Shows the predicted class of the image

Study case

In the testing process were included individuals from the university, in order to get real images in motion from people.

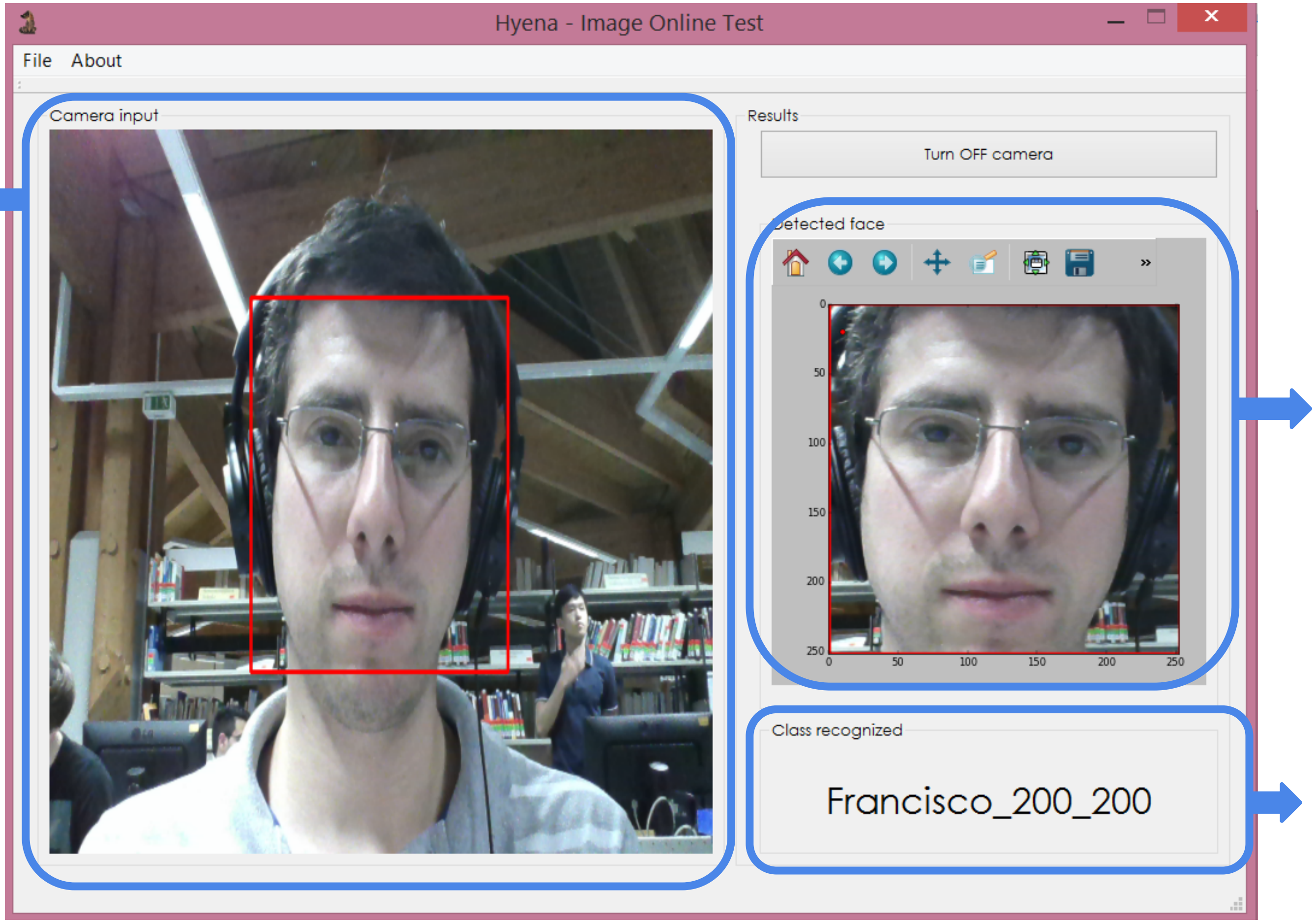
Sometimes the prediction was misleading due to environmental or lighting conditions.

The whole process (record and preprocessing of image, training algorithms and test online) takes between 10~15 minutes.

The frame ratio with all algorithms enable is around 5~10 FPS.

6. Online test module

Real-time camera images, with detection of faces



Class recognized

Francisco_200_200

Preview of detected faces

class prediction

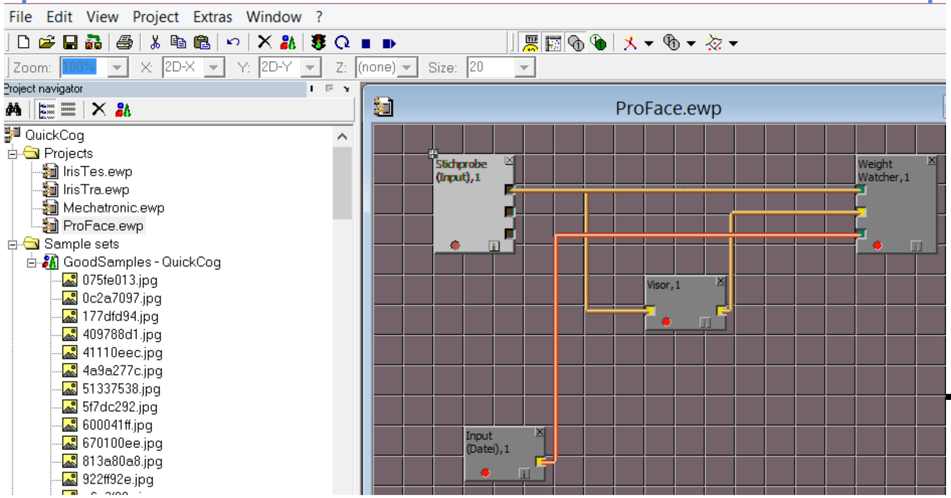
7. QuickCog interface and example

Module’s Characteristics

The applications allows export the image dataset to the follows formats:

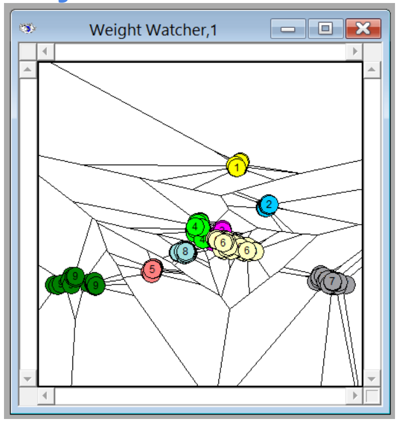
- QuickCog Dataset - .EWS
- Information of classes - NIF

This files can be used for testing purposes and for extending the original scope of Hyena App.



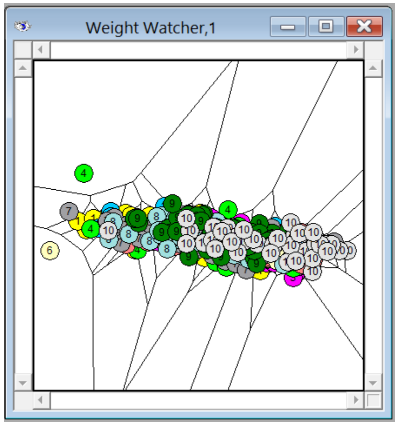
Study case

Separability between classes:



Good samples

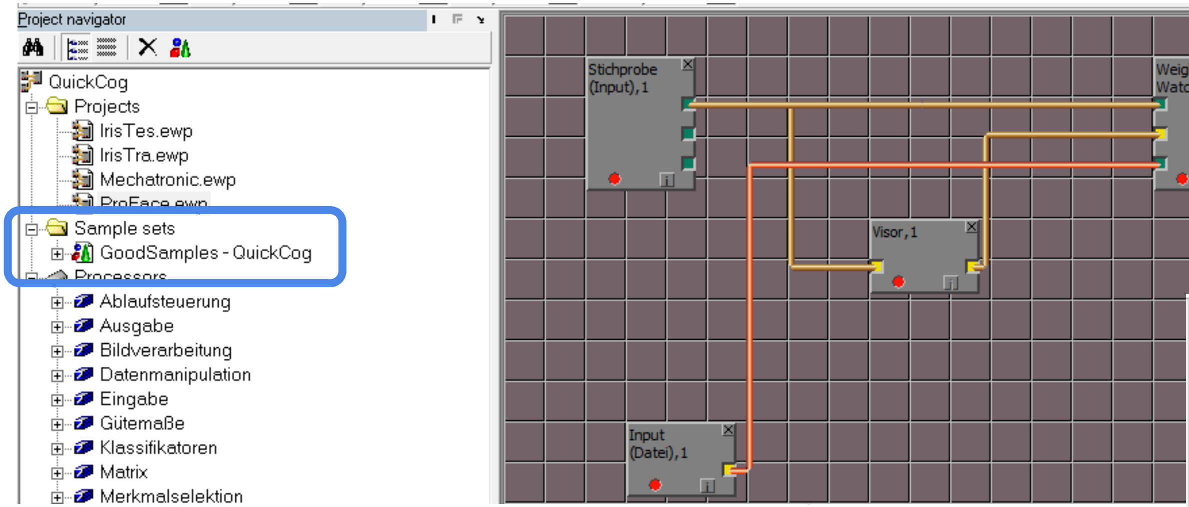
pre processed and filtered



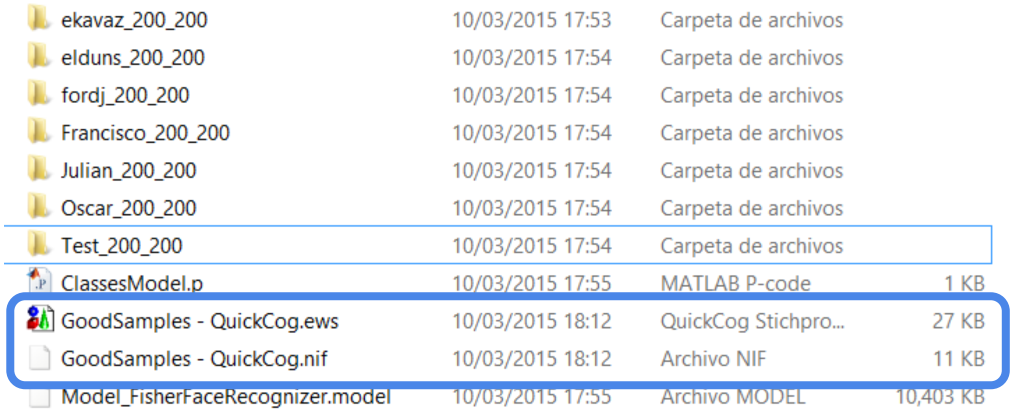
Bad samples:

original images without improvementñ.

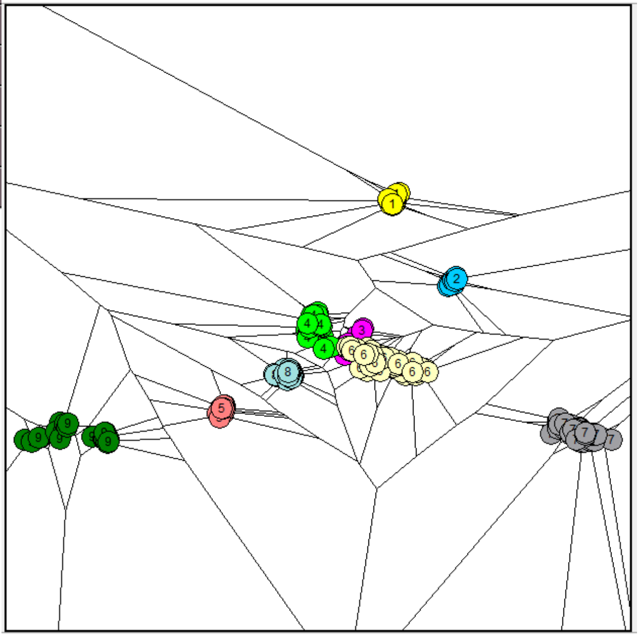
7. QuickCog interface and example



Hyena app export in .EWS and .NIF formats



ekavaz_200_200	10/03/2015 17:53	Carpeta de archivos	
elduns_200_200	10/03/2015 17:54	Carpeta de archivos	
fordj_200_200	10/03/2015 17:54	Carpeta de archivos	
Francisco_200_200	10/03/2015 17:54	Carpeta de archivos	
Julian_200_200	10/03/2015 17:54	Carpeta de archivos	
Oscar_200_200	10/03/2015 17:54	Carpeta de archivos	
Test_200_200	10/03/2015 17:54	Carpeta de archivos	
ClassesModel.p	10/03/2015 17:55	MATLAB P-code	1 KB
GoodSamples - QuickCog.ews	10/03/2015 18:12	QuickCog Stichpro...	27 KB
GoodSamples - QuickCog.nif	10/03/2015 18:12	Archivo NIF	11 KB
Model_FisherFaceRecognizer.model	10/03/2015 17:55	Archivo MODEL	10,403 KB



8. Conclusions

- **General conclusions**

This project has involved too many algorithms and techniques that were discussed previously in the lecture and presents a beta version of a potential commercial application

The hardest part of get features from different sensors, is understand the specific behavior of each signal and discover or select which will be the best algorithms to process it.

- **Acquisition module**

After a long time searching cameras interfaces, we have used OpenCV interface, which offer the best compatibility with different devices and also works multiplatform.

- **Processing module**

There is a lot of different algorithms for working with images, and each of them has a specific target of applications, we have tried to implement the most common algorithms in order to allow a wide range of actions over the images.

8. Conclusions

- **Training module**

We have used algorithms which were already implemented in OpenCV, which offer a better performance and accuracy for working with images, and also take less time in training process.

Working with image have big challenges such as high dimensionality datasets, noise, time of response and hardware limitations (memory)

- **Test online module**

It is very important to cut and clean the face's image, because the algorithms for recognitions are highly depended of the quality of each image, that was the reason for using cascade classificators and different preprocessing steps.

- **QuickCog interface**

This interface allows to compare and extend the capabilities of the Hyena App as well as improve the results for future challenges.

9. Future work

Some extensions were not included in the first version due to time limitations, but we present a list of future improvements of the Hyena App:

- Include more algorithms for preprocessing images
- Add SVM and artificial neural networks algorithms in the training module
- Develop more interfaces (MatLab, Orange, ETC) for extend the capabilities of the project.
- Include more validations tools for testing purposes

Thanks!

Special thanks for supporting this project to:

- Prof. Andreas König
- Msc. Kittikhun Thongpull