

Cover report of bachelor's degree



TEI of Crete
Technological Educational Institute of Crete

Technological Educational Institute of Crete
School of Technological Appliance
Department of Informatics Engineering

Title:

Video Analysis with the use of Java & OpenCV

Author:

Charalampos Polychronakis - 3199

ACKNOWLEDGMENTS

This bachelor thesis is my personal signature to TEI of Crete, where in the years that passed, helped me, academically and personal, to be the person that I am now through meeting friends and colleagues, that are not listed below, for their support and encouragement during my student years.

Firstly, I would like to thank my professor Mr. Costis Aivalis for his help and understanding that showed during the preparation of my thesis.

My colleagues and friends Manolis Ragousis, Stelios Manousakis, Antonis Kardamakis, Manos Panagiotakis, Giorgos Kefalogiannis and Manos Fragkiadakis for helping, supporting and inspiring me to continue my work in my thesis but also as a software engineer.

Most of all, I would like to thank my father and my mother through they do not understand my area they still supporting me.

And last my brother Andreas and my two sisters Eleni and Maria where they always were in my side.

ABSTRACT

Develop Video Analysis program with Open CV and Java

The thesis includes a description and comparative juxtaposition of the techniques and algorithms used to analyze a video. The program that will be built will include interface that will select videos will receive queries on the user's objectives and will allow the export identification and detection of objects and people. Based on OpenCV and the Java language. The search results will be recorded in a database. The searches would be of the type: "How many times selected character appears in a video"

Thesis supervisor: Aivalis Kostis

Title: Lecturer of the Department of Informatics, TEI of Crete



ΣΥΝΟΨΗ

Ανάπτυξη προγράμματος Βίντεο Ανάλυσης με Open CV και Java

Η πτυχιακή περιλαμβάνει περιγραφή και συγκριτική παράθεση των τεχνικών και αλγορίθμων που χρησιμοποιούνται για την ανάλυση ενός βίντεο. Το πρόγραμμα που θα κατασκευαστεί θα περιλαμβάνει interface που θα επιλέγει βίντεο, θα δέχεται ερωτήματα σχετικά με τους στόχους του χρήστη και θα επιτρέπει την εξαγωγή, αναγνώριση και ανίχνευση αντικειμένων και ανθρώπων. Θα βασίζεται στο OpenCV και την γλώσσα Java. Τα αποτελέσματα των αναζητήσεων θα καταχωρούνται σε βάση δεδομένων. Οι αναζητήσεις θα είναι του τύπου: “πόσες φορές εμφανίζεται επιλεγμένος χαρακτήρας σε ένα βίντεο”

Επιβλέπων πτυχιακής: Αϊβαλής Κωστής

Τίτλος : Καθηγητής Εφαρμογών του τμήματος Μηχανικών Πληροφορικής, ΤΕΙ Κρήτης

CONTENTS

ACKNOWLEDGMENTS	1
ABSTRACT	2
ΣΥΝΟΨΗ	4
CONTENTS	5
LIST OF FIGURES	7
1 INTRODUCTION.....	9
1.1 BACKGROUND	9
1.2 THESIS STRUCTURE.....	10
2 THEORY.....	11
2.1 TECHNOLOGIES	11
2.1.1 JAVA	11
2.1.2 OPENCV 3.1.0 / JAVACV.....	11
2.1.3 JDBC	12
2.1.4 VLCj	12
2.1.5 SQLite	13
2.2 STEPS FOR FACE RECOGNITION	13
2.2.1 FACE DETECTION.....	13
2.2.2 FACE RECOGNITION	17
3 IMPLEMENTATION	20
3.1 ENTRY POINT.....	21
3.2 FRAMES/ STAGES	23
3.2.1 PLAYER FRAME.....	23
3.2.2 IMPORT PHOTOS.....	25
3.2.3 IMPORT DATABASES	29
3.2.4 ANALYSIS	31
3.2.5 RESULTS FRAME	37
3.3 CONNECTIVITY DATABASE	39
4 RESULTS	41
4.1 RESULTS EIGENFACE.....	44

4.2 RESULTS FISHERFACE	44
4.3 RESULTS LBPH	45
5 CONCLUSION AND FUTURE WORK	46
5.1 CONCLUSION OF THE COMPARISON OF THE ALGORITHMS	46
5.2 FUTURE WORK	48
6 REFERENCES / BIBLIOGRAPHY	50

LIST OF FIGURES

Figure 1 Template of Front face (parts)	14
Figure 2 Template of face (Complete).....	14
Figure 3 Matching our template with the face of President Obama	14
Figure 4 Haar Cascades.....	15
Figure 5 Using haar cascades in an image	16
Figure 6 An example training set	17
Figure 7 EigenFaces Face Recognizer Principal Components. Source: docs.opencv.org	18
Figure 8 Fisherfaces Face Recognizer Principal Components. Source: docs.opencv.org	19
Figure 9 Dataset after the use of LBPH Source: docs.opencv.org.....	20
Figure 10 Simplified UML Class Diagram - The structure of the project.....	21
Figure 11 Class Diagram - Entry Point	21
Figure 12 FXML of PlayerFrame	22
Figure 13 PlayerFrame Visual Representation (wihout a video).....	23
Figure 14 PlayerFrame visual representation (with a video).....	23
Figure 15 UML of the uses of Player Frame	24
Figure 16 - ImportPhotoFrame Layout.....	25
Figure 17 - Selecting data for the training	26
Figure 18 - Import PhotoForm Filled	26
Figure 19 UML for the PhotoFormController and the rest of the classes we use	28
Figure 20 - Database Entries.....	29
Figure 21 - DBImport UML Class Diagram.....	30
Figure 22 - Analysis Frame.....	31
Figure 23 - Class Diagram for Video Analysis	32
Figure 24 VideoOriginal Class Diagram.....	33
Figure 25 - XML Cascades - LBP on the left HAAR on the right.....	34
Figure 26 VideoAnalysis Class Diagram	36
Figure 27 Result Window/Frame filled with Data	37
Figure 28 PlayerFrame and ResultFrame after the process of Analysis had finished	38
Figure 29 ResultFrame Class Diagram.....	39

Figure 30 Connection Class - Class Diagram	40
Figure 31 Entity and Relationship Model of the database.....	40
Figure 32 VideoAnalysisFrame during the analysis.....	41
Figure 33 Wrong Recognition 1 - Both faces recognized as the same person.....	42
Figure 34 Wrong Recognition 2 - No face was detected.....	43
Figure 35 Wrong Recognition 3- No face recognized.....	43
Figure 36 Wrong Recognition 4 - Wrong recognition of individual	44
Figure 37 Image Details for the second Individual, Maria	47
Figure 38 Image Details for the first Individual, Chari	47
Figure 39 Example of UP & DOWN image	48

1 INTRODUCTION

The purpose of that thesis was to make an application capable to read a video and in the video to try find and recognize a specific person/ individual that user is interested and the application pinpoint, to user, the “location” of that individual on the video. All of these with the use of OpenCV and Java. That was accomplished even with finding some obstacles. Also, one another reason was, to compare and understand the algorithms, for the recognition, and what further development or use of another methods so they can become more accurate in their results.

What wasn't expected, was the fact the Java API of OpenCV is not yet (version 3.1.0 OpenCV) sufficient in some areas like the machine learning part and the video reading (version 3.0.0). Someone with a little knowledge of the field of Machine learning and more knowledge, for example of Web applications and software applications, is expecting Java to be able to do everything. Most of the code of OpenCV is written firstly in C/C++ and secondly in Python. And from that observation anyone can understand that not all programming languages are able to do everything, and always limitations will be exist for each programming language. That obstacle was avoided with the use of Java Native Interface through the use of JavaCV Framework. What was expected through the whole process of recognize an individual was the technical issues. Technical issues in that process is the videos and the images were used to train the algorithms, because the quality of both in some occasions were poor, and as it will be mention later in these thesis, was giving back false results. A small conclusion for future development of anything: **BETTER RESEARCH!!**

In the end, that thesis was successful to create a simple application, with many possibilities to be expand and give more information back to the user. Also someone can say that was the bad thing that thesis doesn't describe fully. And last was able to describe in a good way how the algorithms and the process of recognition are working and make it more.

In the following chapter will be described the knowledge background to accomplish this application. Also, motivation, which was related for the completion of this thesis. Lastly follows a brief overview of the thesis structure and the topics from the following chapters.

1.1 BACKGROUND

"Computer vision is an interdisciplinary field that deals with how computers can be made to gain high-level understanding from digital images or videos. From the perspective of engineering, it seeks to automate tasks that the human visual system can do. Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and in general, deal with the extraction of high-dimensional data from the real world in order to produce numerical or symbolic information, e.g., in the forms of decisions. Understanding in this context means the transformation of visual images (the input of the retina) into descriptions of the world that can interface with other thought processes and elicit appropriate action. This image understanding can be seen as the disentangling of symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory." [1] [2]

In this thesis, we will try to show, through an application, how to acquire, process, analyze and recognize a person, in a video, and after that procedure we will record his/her existence in a database for future use. To be more practical, than theoretical, we will use an open library for computer vision, OpenCV.

Most specifically, OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. OpenCV gives us the ability to read an image, chop it or apply some filters on it, store it, compare it and many other functions.

The applications are:

- 2D and 3D feature toolkits
- Egomotion estimation
- Facial recognition system
- Gesture recognition
- Human–computer interaction (HCI)
- Mobile robotics
- Motion understanding
- Object identification
- Segmentation and recognition
- Stereopsis stereo vision: depth perception from 2 cameras
- Structure from motion (SFM)
- Motion tracking
- Augmented reality

To support some of the above areas, OpenCV includes a statistical machine learning library that contains:

- Boosting
- Decision tree learning
- Gradient boosting trees
- Expectation-maximization algorithm
- k-nearest neighbor algorithm
- Naive Bayes classifier
- Artificial neural networks
- Random forest
- Support vector machine (SVM)

Through this thesis we would like to show a way in which we can automate to recognize a person and record that information as a human would have done in a similar case and determine the value of that knowledge.

1.2 THESIS STRUCTURE

- **Chapter 1**

General information about Computer Vision. OpenCV features which were implement. Then a discussion about the motivation and related work about this thesis

- **Chapter 2**

Theory about the process of recognize the person and the steps we need to follow so to achieve it.

- **Chapter 3**

Analysis of the implementation of the application (VideoAnalyst).

- **Chapter 4**

Contains the results from the VideoAnalyst.

- **Chapter 5**

Contains the conclusion about the bachelor thesis and some thoughts about future optimizations and features.

2 THEORY

In this chapter, we will make a reference in technologies we can take advantage for the development of a Computer Vision application. After that we will analyze the step to achieve Face Recognition.

2.1 TECHNOLOGIES

Some of the techniques we can use are:

- Programming Languages as Java, Python, C++ etc.
- Database Language as MySQL, SQLite, PostgreSQL etc.
- Computer Vision Libraries as OpenCV, JavaCV, Project Oxford, Computer Vision Api (Microsoft), FastCV (Qualcomm) etc.
- Java Technologies/Frameworks as JDBC, VLCj

In our project, we used these technologies:

- Java
- OpenCV / JavaCV
- JDBC
- VLCj

2.1.1 JAVA

Java is a programming language and computing platform first released by Sun Microsystems in 1995. There are lots of applications and websites that will not work unless you have Java installed, and more are created every day. Java is fast, secure, and reliable. From laptops to datacenters, game consoles to scientific supercomputers, cell phones to the Internet, Java is everywhere!

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere"

In this project, java helps to implement different parts as GUI, database use even to use native libraries from C++.

[3]

2.1.2 OPENCV 3.1.0 / JAVACV

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning Application Programming Interface(API) for programming languages like C, C++, Python and Java. OpenCV was worked to give a typical foundation to computer vision applications and to quicken the

utilization of machine learning in the business items. Being a BSD-authorized item, OpenCV makes it simple for organizations to use the code with any methods organizations need.

The library has more than 2500 optimized algorithms which incorporates an exhaustive arrangement of both exemplary and cutting-edge computer vision and machine learning calculations. These calculations can be utilized to distinguish and perceive faces, for instance Viola - Jones we will specify later, distinguish objects, arrange human activities in recordings, track camera developments, track moving objects, extricate 3D models of items, deliver 3D point mists from stereo cameras, fasten pictures together to create a high determination picture of a whole scene, find comparable pictures from a picture database, expel red eyes from pictures taken utilizing streak, take after eye developments, perceive landscape and build up markers to overlay it with enlarged reality, and so forth. OpenCV has more than 47 thousand individuals of client group and evaluated number of downloads surpassing 7 million. The library is utilized widely in organizations, look into gatherings, colleges and by legislative bodies.

Because not all functions are native in Java, to achieve that we use JavaCV. JavaCV uses C++ code of OpenCV with the use of JNI. Presets of commonly used libraries by researchers in the field of computer vision (OpenCV, FFmpeg, libdc1394, PGR FlyCapture, OpenKinect, librealsense, CL PS3 Eye Driver, videoInput, ARToolKitPlus, and flandmark), and provides utility classes to make their functionality easier to use on the Java platform, including Android.

[4]

2.1.3 JDBC

The Java Database Connectivity (JDBC) API is the business standard for database-free availability between the Java programming dialect and an extensive variety of databases SQL databases and other forbidden information sources, for example, spreadsheets or level records. The JDBC API gives a call-level API to SQL-based database get to. The JDBC API gives all inclusive information access from the Java. JDBC innovation likewise gives a typical base on which devices and substitute interfaces can be fabricated.

The JDBC API is involved two bundles:

- java.sql
- javax.sql

You consequently get the two bundles when you download the Java Platform Standard Edition (Java SE) 8.

[5] [6]

2.1.4 VLCj

The VLCj venture is an Open Source venture that gives a Java bindings and an application framework for the VLC media player from the non-benefit association VideoLAN, thus the initials VLC from the media player and j for Java.

The ties can be utilized to fabricate media player customer and server programming utilizing Java - everything from just playing nearby media documents to an out and out video-on-request spilling server is conceivable. A few applications that VLCj is being utilized are to give video capacities to programming being used on oceanographic inquire about vessels and bespoke IPTV and home silver screen arrangements. Additionally, being utilized to make programming for an Open Source camcorder at Elphel and video mapping for the Open Street Map venture. With the utilization of Java API of AWT, Swing JFrame and JavaFX Stage, VLCj enables the client to make a comparative interface with the first application VLC. The system gives the likelihood to see, alter and erase the traits (Frame Rate, Width, Height, Frames Per Second, Length of the video record and so forth.) a video document of any sort of arrangement.

[7]

2.1.5 SQLite

SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. It is a database, which is zero-configured, which means like other databases you do not need to configure it in your system.

SQLite engine is not a standalone process like other databases, you can link it statically or dynamically as per your requirement with your application. SQLite accesses its storage files directly.

[8]

2.2 STEPS FOR FACE RECOGNITION

Two are the main steps to achieve face recognition

- Face Detection
- Face Recognition

2.2.1 FACE DETECTION

To distinguish a face is more straightforward than perceiving a face of a particular individual. To have the ability to verify that a specific picture contains a face (or a few) we should have the ability to characterize the general structure of a face. Fortunately, human appearances don't incredibly contrast from each other; we, as a whole, have noses, eyes, eyebrows, jaw and mouth; and these make the general structure out of a face. [9]

Consider the following 5 figures:

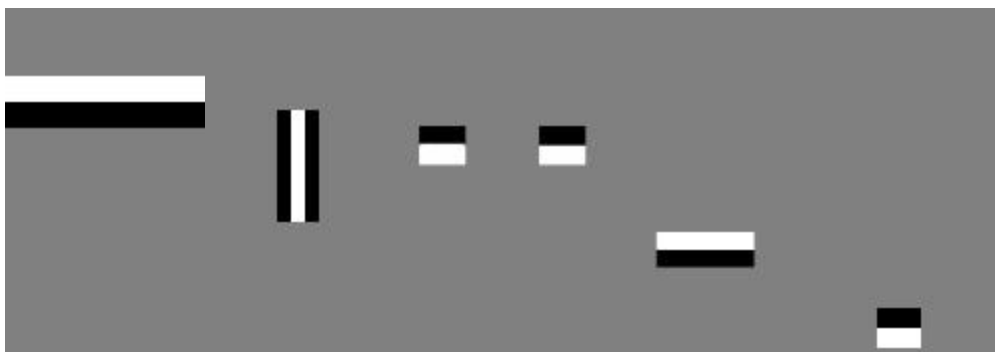


Figure 1 Template of Front face (parts)

Each of these figures speaks to a general element of a human face. Consolidating every one of the features together we, in fact, get something that looks like a face.



Figure 2 Template of face (Complete)

By deciding whether each of these features resembles some piece of our photo, we can determine if the photo contains a face or not. Notice this does not need to be an exact match; we simply need to know whether, generally, each of these features compares to some piece of the picture. The procedure utilized for this reason for existing is Template Matching.

By social affair insights about which such features form faces and how, we can prepare our calculation to utilize the correct features in the correct positions; and along these lines, distinguish faces.

We should see an illustration. Find in the figures beneath how the above features together can be utilized to recognize a face (in particular, the face of President Barack Obama).



Figure 3 Matching our template with the face of President Obama

For this procedure be brisk, we outline it such that we initially check the coarse features which speak to the coarse structure of a face; and just if these features coordinate, we proceed to the following emphasis and utilize better features. In each such emphasis, we can rapidly dismiss ranges of the photo which don't coordinate a face, and continue checking those which we don't know about. In each cycle, we increment the conviction that the checked territory is without a doubt a face, until at last we stop and make our assurance.

As such, instead of deciding whether the picture contains a face, we would more be able to rapidly decide whether the picture does not contain a face; since disposals should be possible rapidly, while

acknowledgment of appearances will require additional time. We call such a procedure a falling procedure.

The strategy portrayed here is an over-disentangled depiction of the Viola-Jones technique (also known as Haar cascades). [10]

Viola – Jones Method or Object Detection utilizing Haar include based course classifiers is a viable question identification strategy proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a course work is prepared from a great deal of positive and negative pictures. It is then used to identify protests in different pictures.

Here we will work with confront recognition. At first, the calculation needs a considerable measure of positive (pictures of countenances) and negative (pictures without faces) to prepare the classifier. At that point, we have to remove characteristics from it. For this, haar characteristics appeared in underneath picture are utilized. They are much the same as our convolutional bit. Each component is a solitary esteem got by subtracting total of pixels under white rectangle from whole of pixels under dark rectangle.

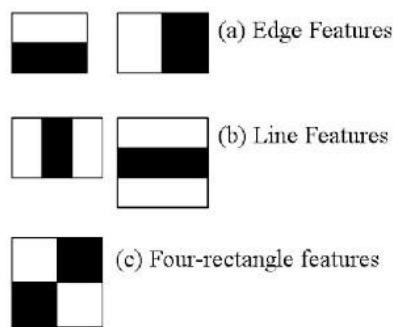


Figure 4 Haar Cascades

Presently all conceivable sizes and areas of every piece is utilized to ascertain a lot of characteristics. (Simply envision what amount of calculation it needs? Indeed, even a 24x24 window comes about more than 160000 characteristics). For each component computation, we have to discover total of pixels under white and dark rectangles. To tackle this, they presented the indispensable pictures. It disentangles count of entirety of pixels, how expansive might be the quantity of pixels, to an operation including only four pixels. Pleasant, would it say it isn't? It influences things super-to quick.

However, among every one of these characteristics we ascertained, a large portion of them are insignificant. For instance, consider the picture underneath. Top line demonstrates two great characteristics. The main component chose appears to concentrate on the property that the district of the eyes is frequently darker than the area of the nose and cheeks. The second element chose depends on the property that the eyes are darker than the scaffold of the nose. However, similar windows applying on cheeks or some other place is superfluous. So how would we choose the best characteristics out of 160000+ characteristics? It is accomplished by Adaboost. [11]

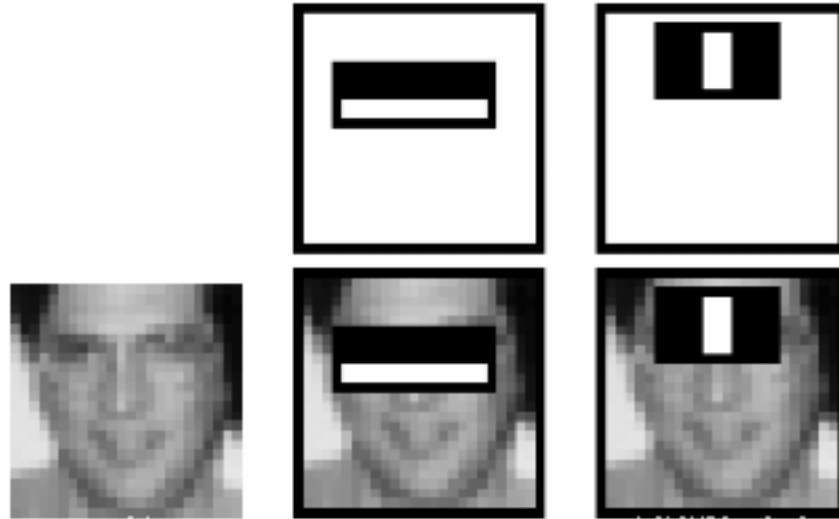


Figure 5 Using haar cascades in an image

For this, we apply every single element on all the preparation pictures. For each component, it finds the best limit which will group the appearances to positive and negative. Be that as it may, clearly, there will be blunders or misclassifications. We select the characteristics with least mistake rate, which implies they are the characteristics that best orders the face and non-confront pictures. (The procedure is not as basic as this. Each picture is given an equivalent weight first and foremost. After every order, weights of misclassified pictures are expanded. On the other hand same process is finished. New blunder rates are ascertained. Likewise new weights. The procedure is proceeded until the point that required precision or mistake rate is accomplished or required number of characteristics are found).

Last classifier is a weighted entirety of these frail classifiers. It is called feeble in light of the fact that only it can't arrange the picture, however together with others shapes a solid classifier. The paper says even 200 characteristics furnish identification with 95% precision. Their last setup had around 6000 characteristics. (Envision a lessening from 160000+ characteristics to 6000 characteristics. That is a major pick up).

So now you take a picture. Take each 24x24 window. Apply 6000 characteristics to it. Check on the off chance that it is confront or not. Stunning. Amazing. Is it accurate to say that it isn't somewhat wasteful and tedious? Indeed, it is. Creators have a decent answer for that.

In a picture, the greater part of the picture area is non-confront district. So it is a superior plan to have a basic technique to check if a window is not a face district. In the event that it is not, dispose of it in a solitary shot. Try not to process it once more. Rather concentrate on district where there can be a face. Along these lines, we can discover more opportunity to check a conceivable face district.

For this they presented the idea of Cascade of Classifiers. Rather than applying all the 6000 characteristics on a window, assemble the characteristics into various phases of classifiers and apply one-by-one. (Regularly initial couple of stages will contain less number of characteristics). On the off chance that a window falls flat the main stage, dispose of it. We don't consider remaining characteristics on it. In the event that it passes, apply the second phase of characteristics and proceed with the procedure. The window which passes all stages is a face locale. How is the arrangement!!!

Creators' indicator had 6000+ characteristics with 38 phases with 1, 10, 25, 25 and 50 includes in initial five phases. (Two characteristics in the above picture is really gotten as the best two characteristics from Adaboost). As per creators, on a normal, 10 includes out of 6000+ are assessed per sub-window.

So this is a basic natural clarification of how Viola-Jones confront identification functions. Read paper for more points of interest or look at the references in Additional Resources area. [12, p. 1]

2.2.2 FACE RECOGNITION

The contrast between confront discovery and acknowledgment is that in identification we simply need to decide whether there is some face in the picture, however in acknowledgment we need to decide whose face it is. In the above illustration we identified a face, which we perceive as President Obama.

With a specific end goal to comprehend the strategies for perceiving faces, further developed numerical learning is required; to be specific direct polynomial math and insights.

OpenCV gives three techniques for confront acknowledgment:

- i. Eigenfaces
- ii. Fisherfaces
- iii. Local Binary Patterns Histograms (LBPH).

Every one of the three techniques play out the acknowledgment by contrasting the face with be perceived with some preparation set of known countenances. In the preparation set, we supply the calculation faces and instruct it to which individual they have a place. At the point when the calculation is made a request to perceive some obscure face, it utilizes the preparation set to make the acknowledgment. Each of the three previously mentioned strategies utilizes the preparation set a bit in an unexpected way.

Eigenfaces and Fisherfaces locate a numerical depiction of the most overwhelming characteristics of the preparation set all in all. LBPH breaks down each face in the preparation set independently and autonomously. [13]



Figure 6 An example training set

i. Eigenfaces

In that sense, we would be centering on the regions of the face with the most extreme change. To example, from eyes to the nose there is a critical change. Also, same applies starting with the nose to the mouth. The point when you take a gander at various faces, you analyze them by taking a gander at these areas, in light Toward getting the greatest variety Around faces, they help you separate particular case face from the opposite.

In this way, may be how EigenFaces recognizer meets expectations. It takes a gander whatsoever those preparing pictures of every last one of individuals all in all and tries on extricate those parts which are applicable Also suitable What's more discards whatever remains. These critical features are known as vital parts.

The following is a picture demonstrating those difference concentrated starting with An rundown for countenances.

These eigenvectors represent the most prominent features of the dataset, and since we talk about face images, the eigenvectors actually represent the strongest characteristics of the faces in the dataset. See for example the first 8 eigenvectors of the dataset from above:

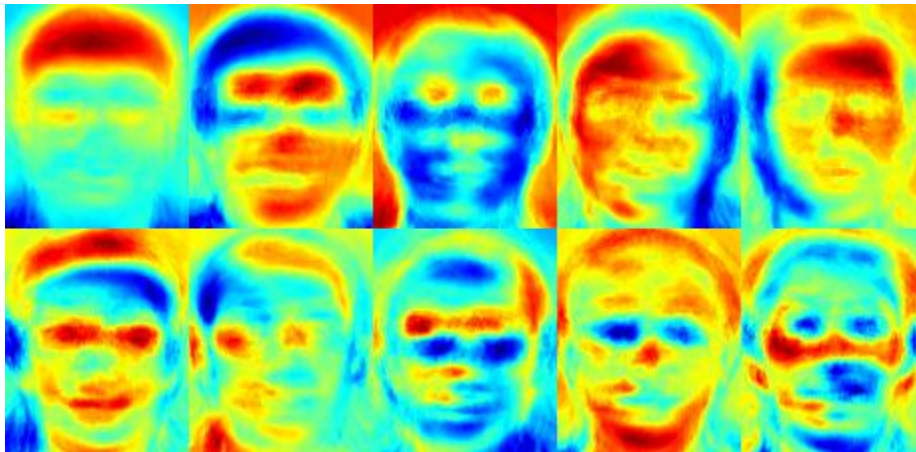


Figure 7 EigenFaces Face Recognizer Principal Components. Source: docs.opencv.org

Now, whenever we are provided with a new, unknown, face, we can decompose it to the basis we found, see which eigenvector(s) “explain” most of the face, and thus determine to which person it belongs. [14] [13] [10]

ii. Fisherfaces

This algorithm is an enhanced form of the last one. As we just observed, Eigenfaces takes a gander at all the preparation countenances of the considerable number of individuals on the double and discovers main segments from every one of them joined. By doing that, it doesn't concentrate on the highlights that segregate one individual from another. Rather, it focuses on the ones that speak to every one of the characteristics of the considerable number of individuals in the preparation information, in general. Since EigenFaces additionally discovers light as a helpful part, it will discover this variety exceptionally pertinent for confront acknowledgment and may dispose of the highlights of the other individuals'

countenances, thinking of them as less valuable. At last, the difference that Eigenfaces has separated speaks to only one person's facial highlights.

How to settle the issue? We can do it by tuning Eigenfaces with the goal that it removes valuable highlights from the characteristics of every individual independently as opposed to extricating them from every one of the countenances consolidated. Along these lines, regardless of the possibility that one individual has high enlightenment transforms, it won't influence the other individuals' highlights extraction process. Correctly, FisherFaces confront recognizer calculation separates primary segments that separate one individual from the others. In that sense, a person's segments don't overwhelm (turn out to be more helpful) over the others.

The following is a picture of essential parts utilizing Fisherfaces calculation.

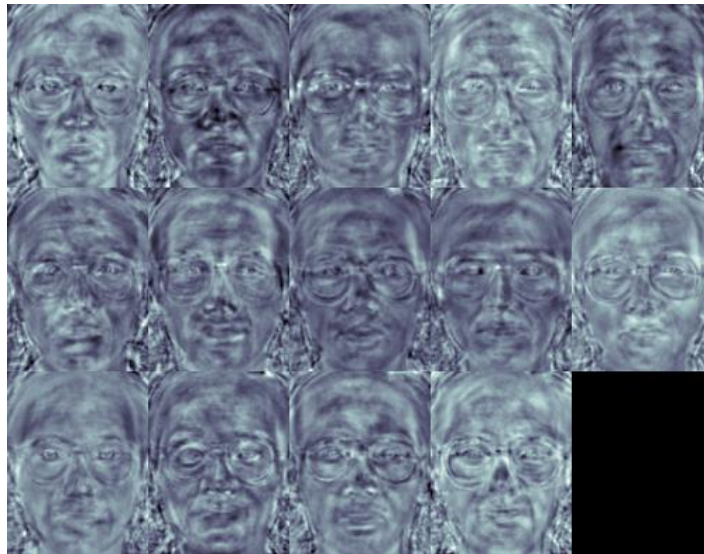


Figure 8 Fisherfaces Face Recognizer Principal Components. Source: docs.opencv.org

You can see that the features represent faces which receive the name of Fisher Faces. Are you noticing a theme with the names of the algorithms?

One thing to note here is that Fisherfaces only prevents features of one person from becoming dominant, but it still considers illumination changes as a useful feature. We know that light variation is not a useful feature to extract as it is not part of the actual face. [14] [13] [10]

iii. Local Binary Patterns Histogram (LBPH)

The LBPH method takes a different approach than the eigenfaces method. In LBPH each images are analyzed independently, while the eigenfaces method and fisherfaces method, looks at the dataset as a whole. The LBPH method is somewhat simpler, in the sense that we characterize each image in the dataset locally, and when a new unknown image is provided, we perform the same analysis on it and compare the result to each of the images in the dataset. The way which we analyze the images is by characterizing the local patterns in each location in the image.

Following is such a local binary patterns analysis on each of the images of the dataset from above:



Figure 9 Dataset after the use of LBPH Source: docs.opencv.org

With LBP it is possible to describe the texture and shape of a digital image. This is done by dividing an image into several small regions from which the features are extracted. These features consist of binary patterns that describe the surroundings of pixels in the regions. The obtained features from the regions are concatenated into a single feature histogram, which forms a representation of the image. Images can then be compared by measuring the similarity (distance) between their histograms. According to several studies face recognition using the LBP method provides very good results, both in terms of speed and discrimination performance. [14] [13] [10]

3 IMPLEMENTATION

In this chapter, we will focus in the technical details about the application and we will explain extensively the most important ones and the use of them.

The structure of the project is:

- Entry Point
- Frames / Stages
 - Player Frame
 - Import Photos
 - Import Database
 - Analysis

- Video Original
- Video Face Recognition
- Face Recognizer
- Results Frame
- Connectivity Database

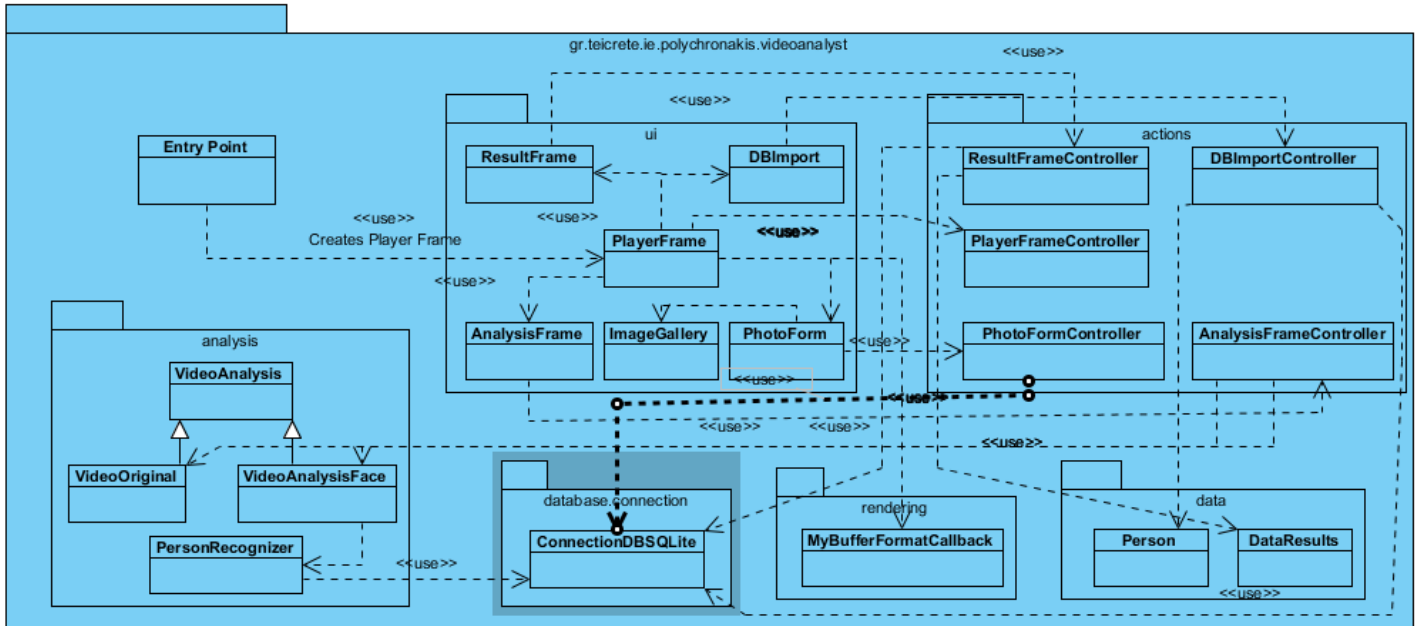


Figure 10 Simplified UML Class Diagram - The structure of the project

3.1 ENTRY POINT

In this part is where the application starts.

Here the “main” Class extends the Application class which is class from the JavaFX libraries.

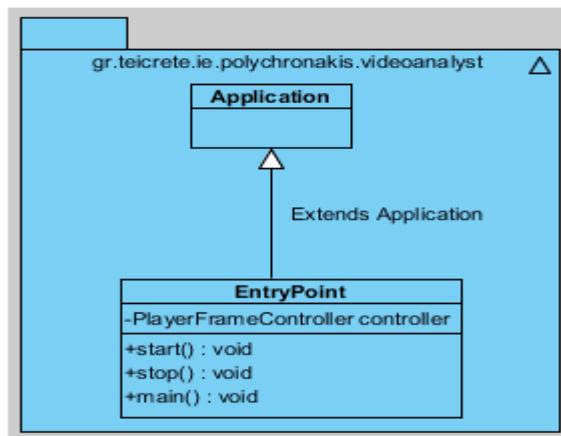


Figure 11 Class Diagram - Entry Point

More specific:

The entry point for JavaFX applications is the Application class. The JavaFX runtime does the following, in order, whenever an application is launched:

1. Constructs an instance of the specified Application class
2. Calls the `init()` method
3. Calls the `start(javafx.stage.Stage)` method
4. Waits for the application to finish, which happens when either of the following occur:
 - a. the application calls `Platform.exit()`
 - b. the last window has been closed and the `implicitExit` attribute on `Platform` is true
5. Calls the `stop()` method

Note that the `start` method is abstract and must be overridden. The `init` and `stop` methods have concrete implementations that do nothing. [15]

In this class, also we do:

A component that uses discovery strategies to locate the libvlc native libraries and register their location with the JNA run-time. To try and locate automatically the libvlc native libraries we only need to create an instance of the `NativeDiscovery` and call the `discover()` method.

Because we want to separate the application design from the application logic we will use the FXML files. FXML files are XML based scripts that allow to define the design of an application with the use of tags like XML. In `FXMLLoader` we load the fxml with the components are written inside on it.

A small glimpse how an fxml is written like

```

1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.scene.canvas.Canvas?>
4 <?import javafx.scene.control.Button?>
5 <?import javafx.scene.control.Label?>
6 <?import javafx.scene.control.Menu?>
7 <?import javafx.scene.control.MenuBar?>
8 <?import javafx.scene.control.MenuItem?>
9 <?import javafx.scene.control.Separator?>
10 <?import javafx.scene.control.SeparatorMenuItem?>
11 <?import javafx.scene.control.Slider?>
12 <?import javafx.scene.control.ToolBar?>
13 <?import javafx.scene.layout.BorderPane?>
14 <?import javafx.scene.layout.HBox?>
15 <?import javafx.scene.layout.VBox?>
16
17 <VBox prefHeight="85.0" prefWidth="700.0" xmlns="http://javafx.com/javafx/8.0.60" xmlns:fx="http://javafx.com/fxml/1" fx:controller="gr.teicrete.ie.polychr...
18 <children>
19   <MenuBar VBox.vgrow="NEVER">
20     <menus>
21       <Menu mnemonicParsing="false" text="File">
22         <items>
23           <MenuItem fx:id="open" mnemonicParsing="false" onAction="#onClickOpenVideo" text="Open Video..." />
24           <!--
25           <Menu mnemonicParsing="false" text="Open Recent Video" />
26           -->
27           <SeparatorMenuItem mnemonicParsing="false" />
28           <MenuItem fx:id="importPBTN" mnemonicParsing="false" onAction="#onClickIP" text="Import Photo from PC" />
29             <MenuItem fx:id="importDB" mnemonicParsing="false" onAction="#onClickDB" text="Import Photo from Database" />
30           <!--
31           <SeparatorMenuItem mnemonicParsing="false" />
32           <MenuItem mnemonicParsing="false" text="Save" />
33           <MenuItem mnemonicParsing="false" text="Save As..." />
34           -->
35           <SeparatorMenuItem mnemonicParsing="false" />

```

Figure 12 FXML of PlayerFrame

After we load the FXML file we need to create the root Parent node which will be assigned later to the scene. Parent Node here is the component that contains the rest of components inside of it. That Node for the first Frame is VBox component.

As we mention before, with separate the application design with application logic, we can use css files which are responsible for the design and makes our application able to play even in web form or in a smartphone. After we do these two steps we add the scene we created to our stage. We also with the use of the loader we make a reference of the ResultsFrameController so the project to be able to initialize the mainStage with the use of the `init()` method. The same procedure is for each frame which is going to be created with small changes that are need it for each frame individually.

3.2 FRAMES/ STAGES

In this section, we will refer to each frame individually and how are connected between them and what other classes they are using for.

3.2.1 PLAYER FRAME

PlayerFrame is the mainly frame and responsible for connected the other frames together and their information.

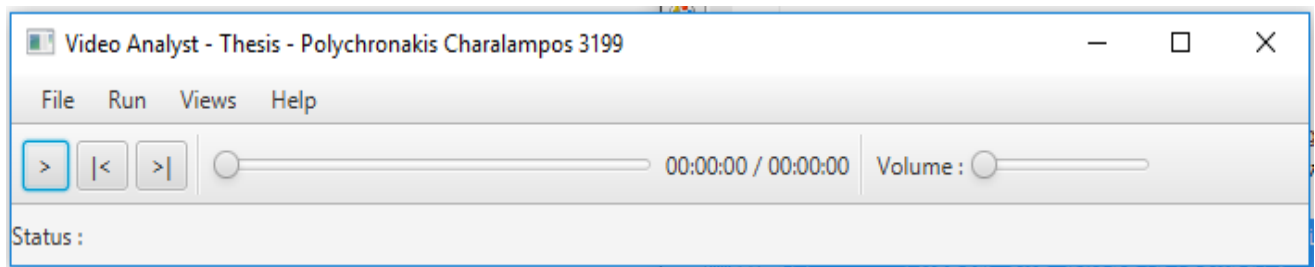


Figure 13 PlayerFrame Visual Representation (without a video)

With the use of the VLCj framework, developers can create a similar media player like VLC for the needs of the application concerned the videos.

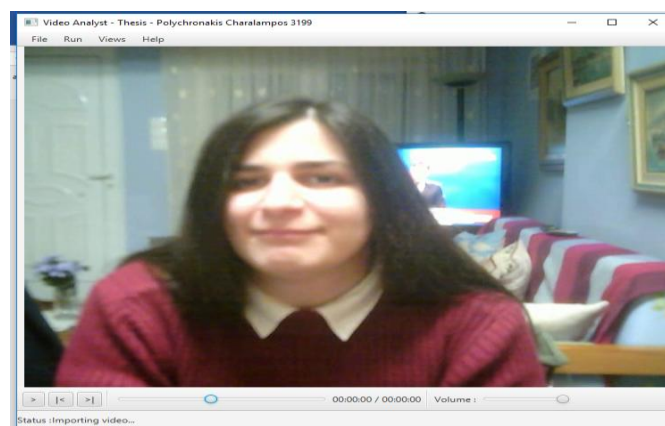


Figure 14 PlayerFrame visual representation (with a video)

The structure of the class is the following.

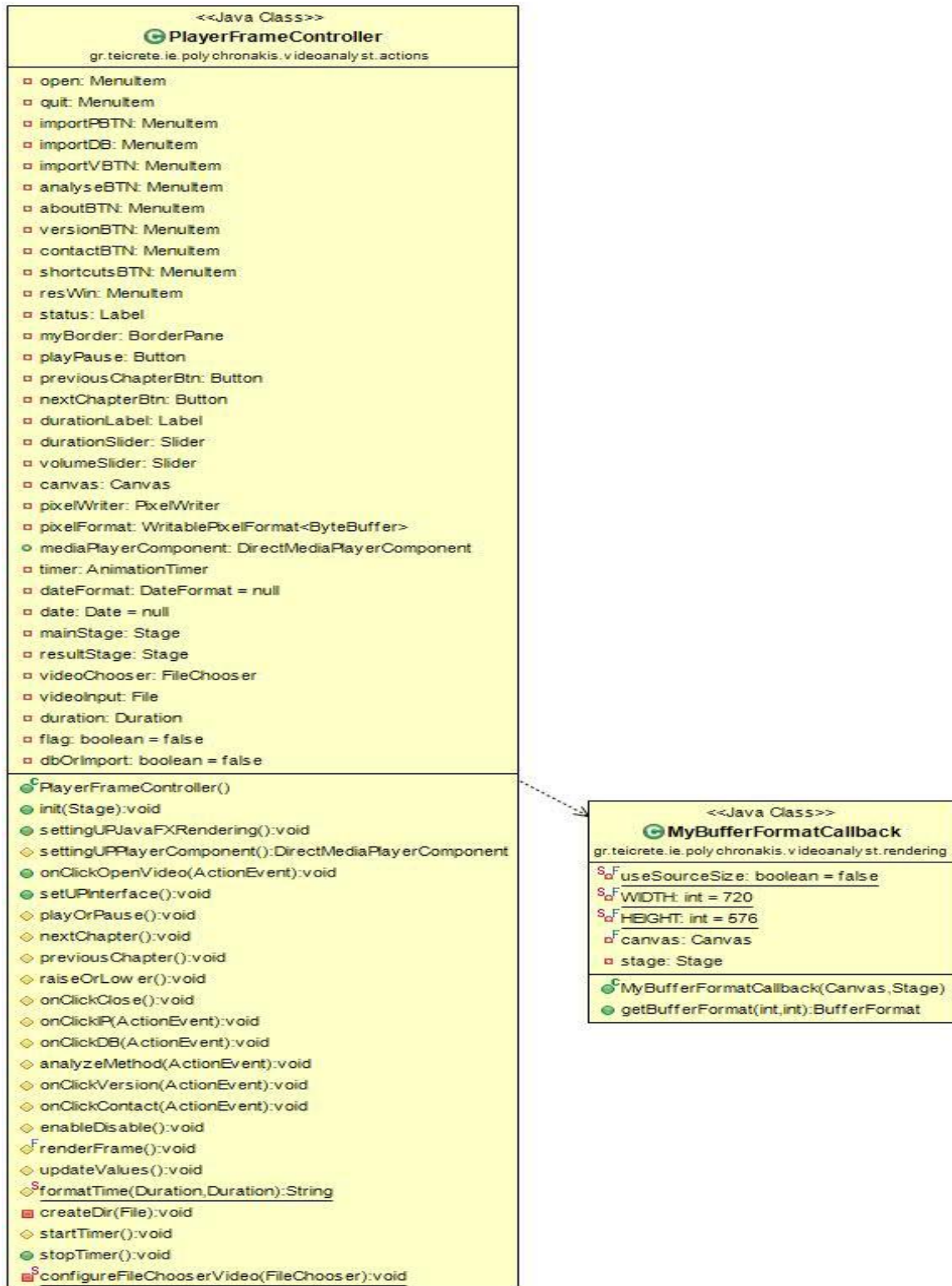


Figure 15 UML of the uses of Player Frame

Let's describe in short how the application works. As a first step to use this application is to load the video from the computer. Second step is to pass the photos, that applications needs, to create a training set which is also going to be stored for future use in the database. Also, if they application has been used in the past, there is the possibility to the user retrieve photos from the database instead load again the same photos for the same individual. This dataset gives the ability to the application to learn who he/she is the individual is looking for. Now the application has all the information it needs, so it can analyze and create more information for the individual it is seeking on the video (moment of appearance of the individual). In the end, the application gives back to user, the new video with the information that has found from the analysis process, and a table with the all the information that found.

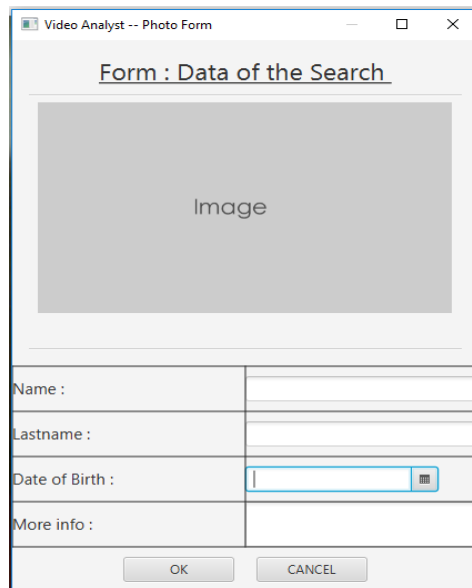
From all the process described above, in the Player Frame we are loading the video and the photos or taking information from the database and initiate the analysis process so later to receive the final data. That frame is also responsible to read, pass, analyze and print the information back to user with the use of the other frames. In the next sections, is described what information each frame receives, where is called, and how each frame handles that information.

3.2.2 IMPORT PHOTOS

After user select,

File -> Import from PC

he/she will get the ImportPhotoFrame. This Frame is important because it helps to record the individual, in the database, the user wants and to create a dataset that will be used later, or for future use also, in the analysis to train the program.




Form : Data of the Search	
Image	
Name :	<input type="text"/>
Lastname :	<input type="text"/>
Date of Birth :	<input type="text"/> 
More info :	<input type="text"/>
<input type="button" value="OK"/> <input type="button" value="CANCEL"/>	

Figure 16 - ImportPhotoFrame Layout

In the following pictures, the reader can see step by step how can any user import photos/pictures in the application.

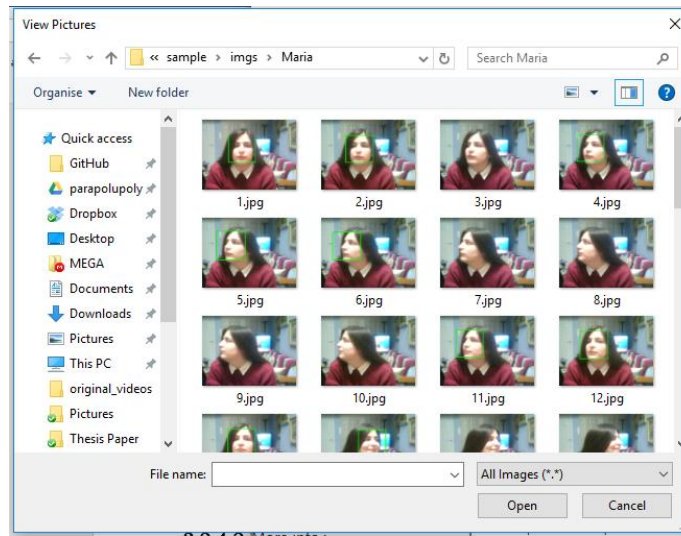


Figure 17 - Selecting data for the training

In the below picture, we see only the face part from the rest of a picture which the algorithm needs to create the dataset and after use that dataset to be trained.

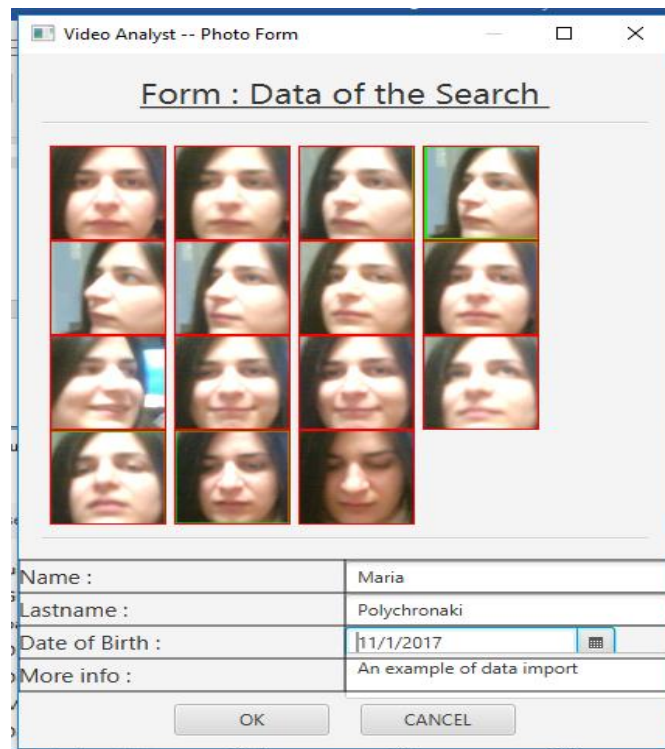


Figure 18 - Import PhotoForm Filled

Here is good to notice, that the application needs more than 10 images because if later to the training algorithm receives only one photo it will not be accurate in the recognition of the person that user wants.

In the following picture, is a representation of the PhotoFormController in UML and the connection with PlayerFrameController has been discussed before.

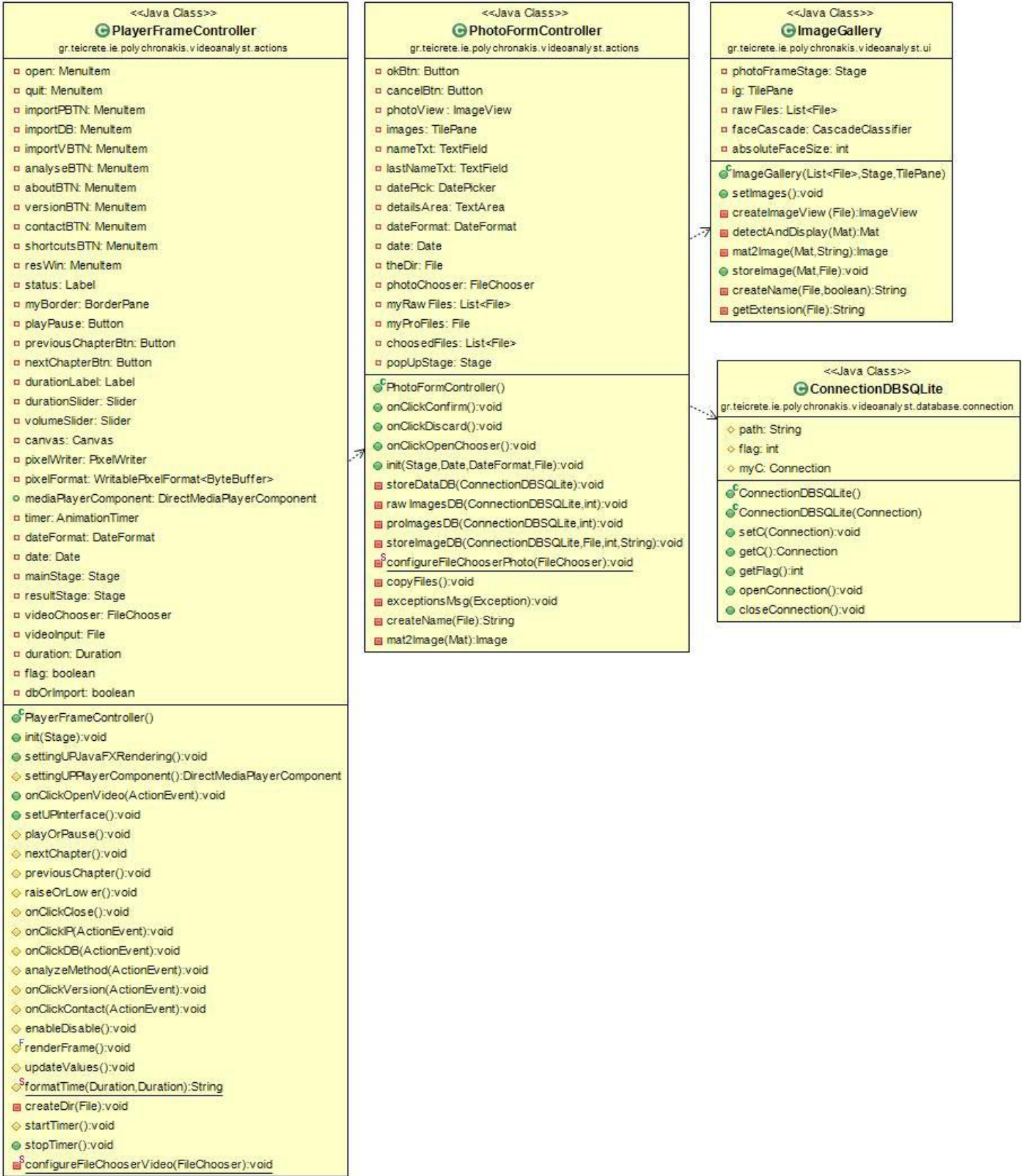


Figure 19 UML for the PhotoFormController and the rest of the classes we use

Here is the connection with the Database with the PlayerFrameController class. The ImageGallery class create the thumbnails from the importing pictures, only the face from the picture, (with the use of Viola-Jones) and to represent it to the user.

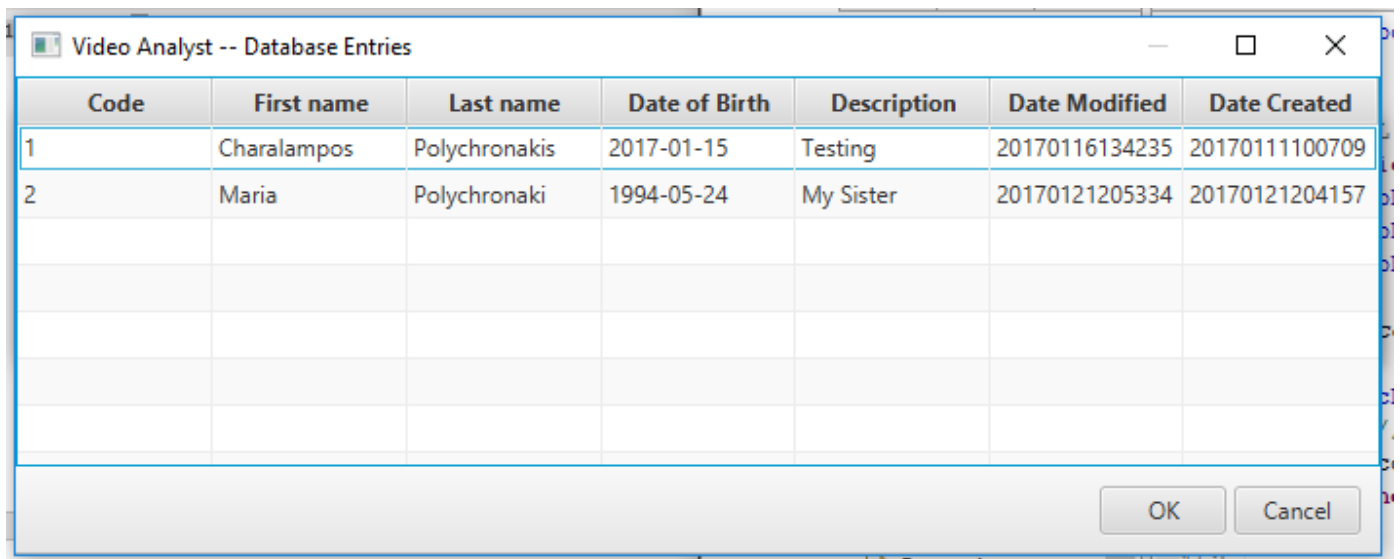
3.2.3 IMPORT DATABASES

User can select,

File -> Import from Database

and he/she will get the Database Entries Frame / DBImportFrameController.java

Because of the use of a database, for storing previous results or datasets, in that frame the user is able to see older records for persons that used before in older analysis.



Code	First name	Last name	Date of Birth	Description	Date Modified	Date Created
1	Charalampos	Polychronakis	2017-01-15	Testing	20170116134235	20170111100709
2	Maria	Polychronaki	1994-05-24	My Sister	20170121205334	20170121204157

Figure 20 - Database Entries

In that form are the entries to our database like an Excel Spreadsheet. For now the fields are used as an example what fields any user want to have, based on the use of the application we can change they field. Later in the conclusion section are some examples what use of the current application can be used with of course some changes.

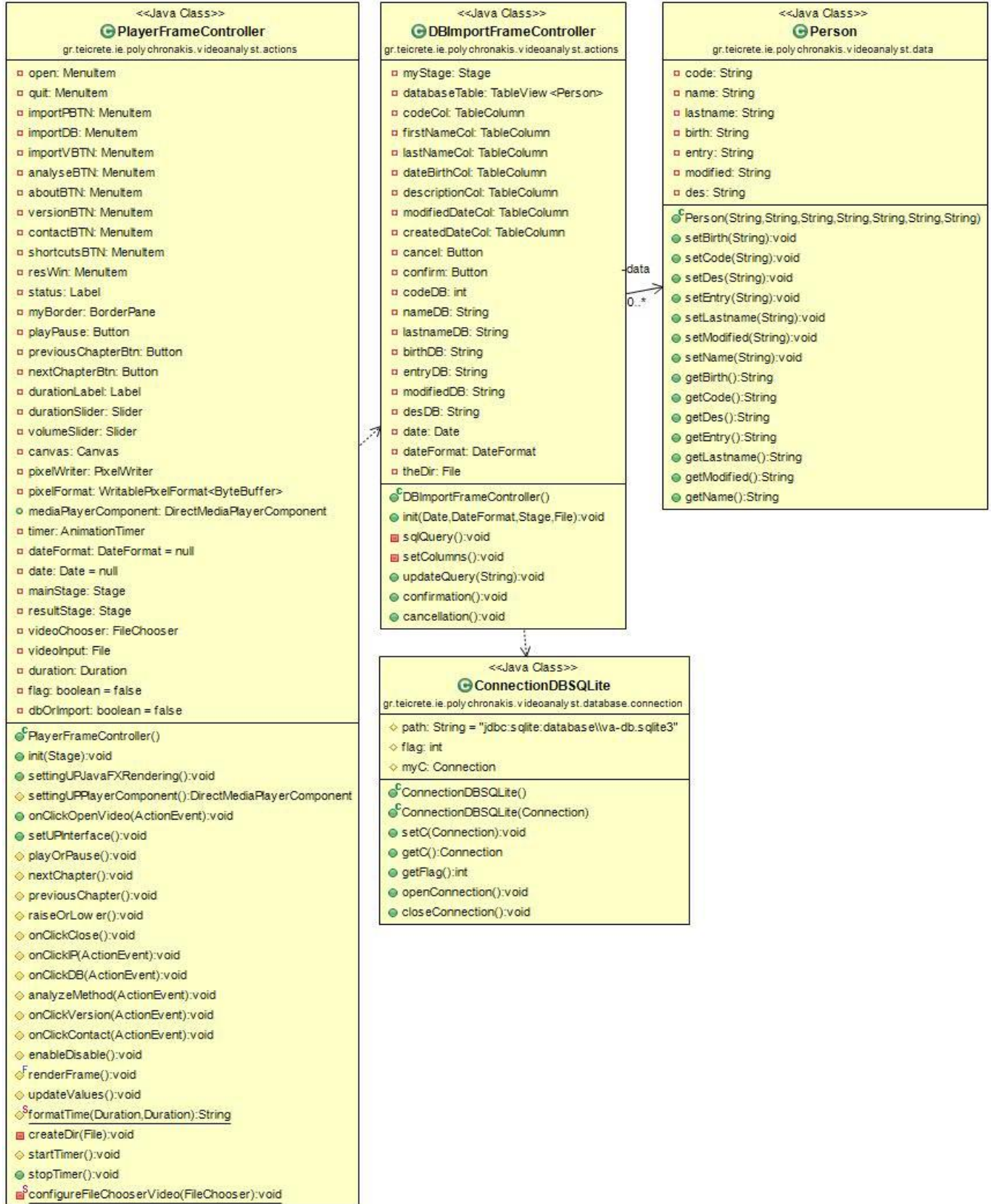


Figure 21 - DBImport UML Class Diagram

This class diagram, shows the connection with our main class (PlayerFrameController). Also shows the association with the database and also a class named Person. The class Person is been used from the object TableView databaseTable and matches the attributes from the class Person with the same Column (ex. Person.code = codeCol).

3.2.4 ANALYSIS

Select,

Run -> Analyze

so, the process of the analysis can start. It will appear the frame below with the data that user has imported in the application.

The most important part of the project was to be able to recognize a person. In the next sections, is described how the face-detection, recognition and store the exact appearance of the individual based the frame who appeared.

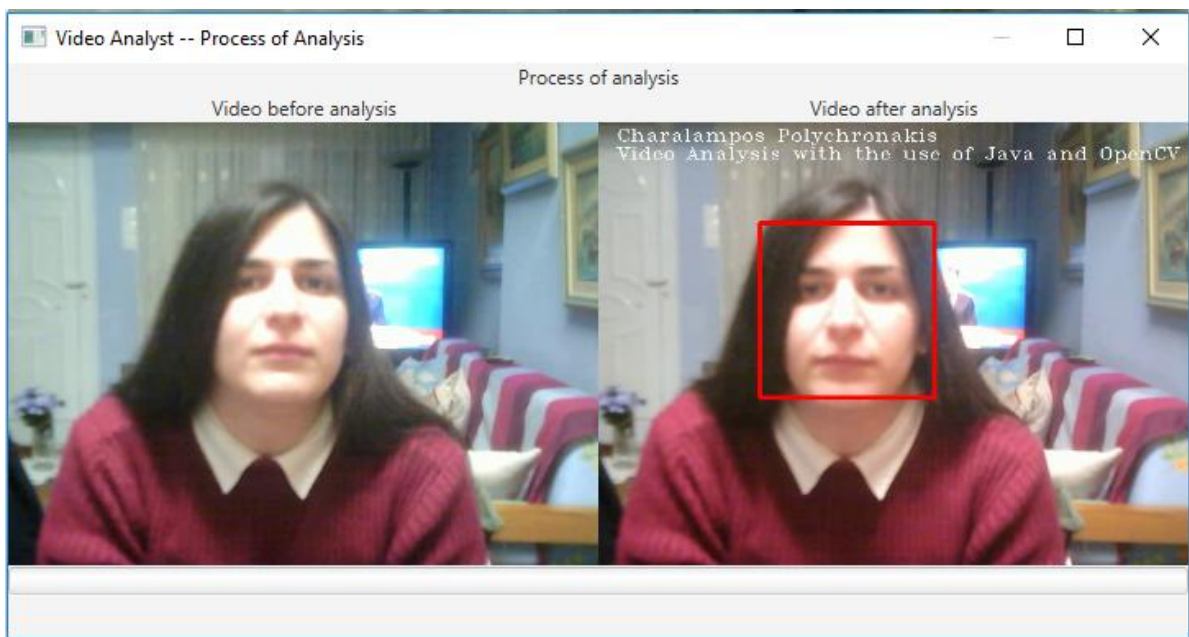


Figure 22 - Analysis Frame

The most important frame because we can compare before and after the analysis we do in our video before the process ends and we move in the results frame.

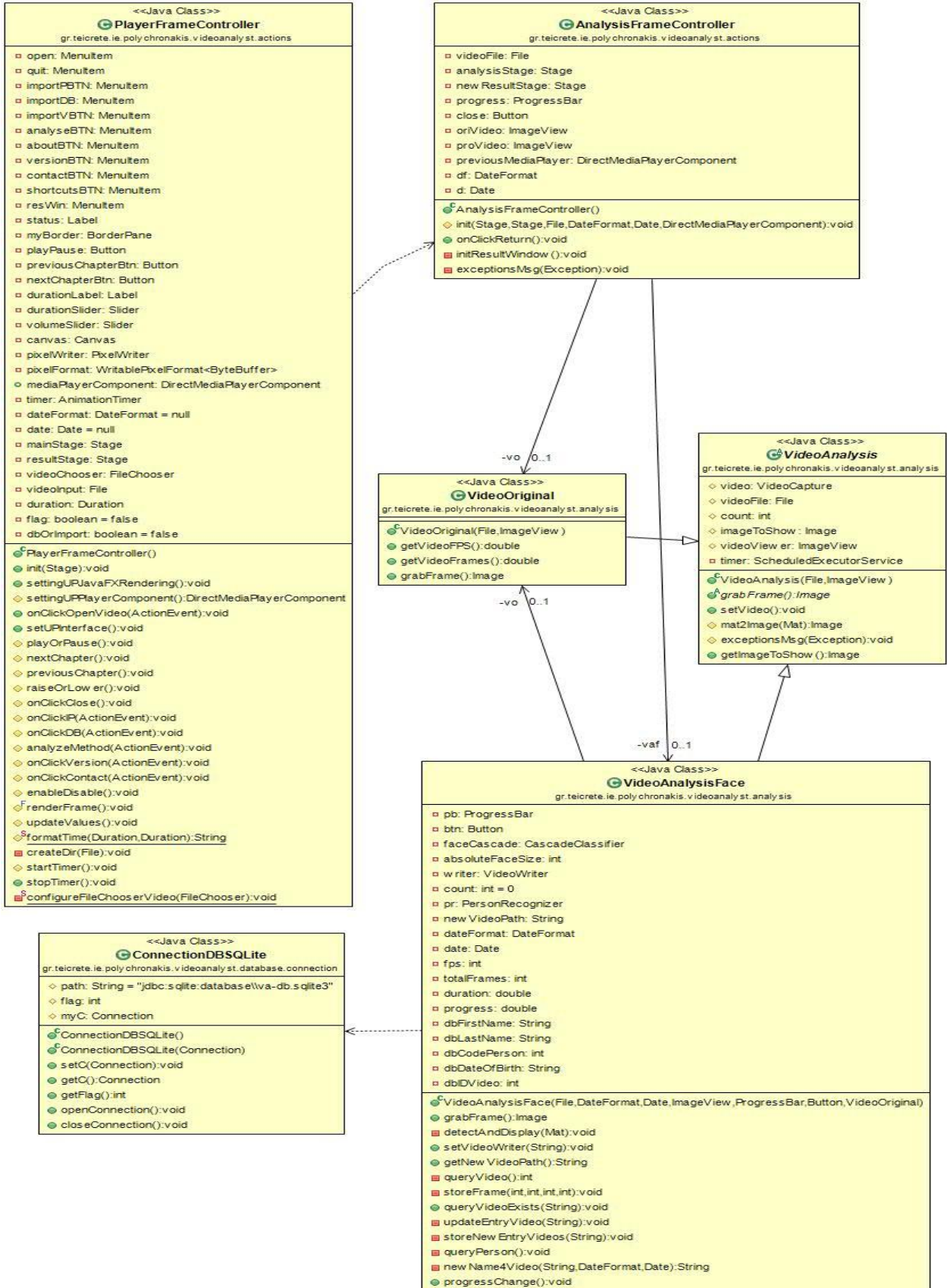


Figure 23 - Class Diagram for Video Analysis

The AnalysisFrameController is responsible for the UI part in our application but also creates instances for the VideoOriginal and VideoAnalysisFace, which both classes inherit from the VideoAnalysis Class.

VideoAnalysis gives the basic structure how to display a frame in the AnalysisFrameController regardless the processes of the frames by its subclasses.

In the next sub-chapters, I will describe the rest classes from the above UML Class Diagram.

i. VIDEO ORIGINAL

Here, in that class, VideoAnalysis provides the common attributes in the classes VideoOriginal and VideoAnalysisFace as mentioned before. The purpose of VideoOriginal is to create an Image object named imageToShow from the original video, that user had gave, so to be able to see in the AnalysisFrame a preview from before and after of the analysis process. The reason we have such special class here is only because a video is a series from continues pictures/frames, and this class is extracting that picture from each frame from the video and displays it back to user from the AnalysisFrame. That process is coded in the grabFrame method, which is an abstract method in VideoOriginal class. Also, is good to mention that OpenCV cannot process a picture. OpenCV translates the picture (a JPEG,GIF,TIFF... etc. file) in a type of two-dimensional Matrix. More of that process is going to be described in VideoAnalysis where is going to be also the process of the recognition of the person.

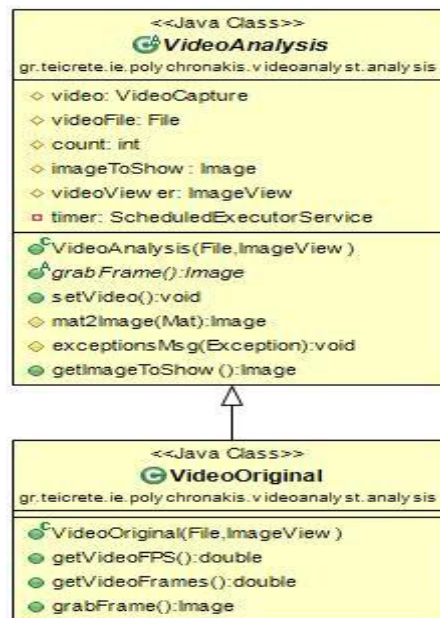


Figure 24 VideoOriginal Class Diagram

ii. VIDEO FACE RECOGNITION

VideoAnalysisFace is the most interesting class in that application. In the previous chapter is mentioned the theory for face detection and after the face recognition. In that class, the reader is able to see the theory to become practice. As in the theoretical part, here is also the same idea with some more steps to achieve it. Likewise, in class VideoOriginal, after the video is loaded the application start extracting frames. Before the application moves to the next frame, starts to detect face. The code for that is on the method detectAndDisplay. As first step of this method is to choose the type of Cascade. As mention before cascade helps to detect the face by comparing each of the characteristics (nose, ears, mouth, etc.) with the part of the frame where a face exists. In the application, we have two types of cascades, haar-cascades and LBP-cascades (Local Binary Patterns). [16]

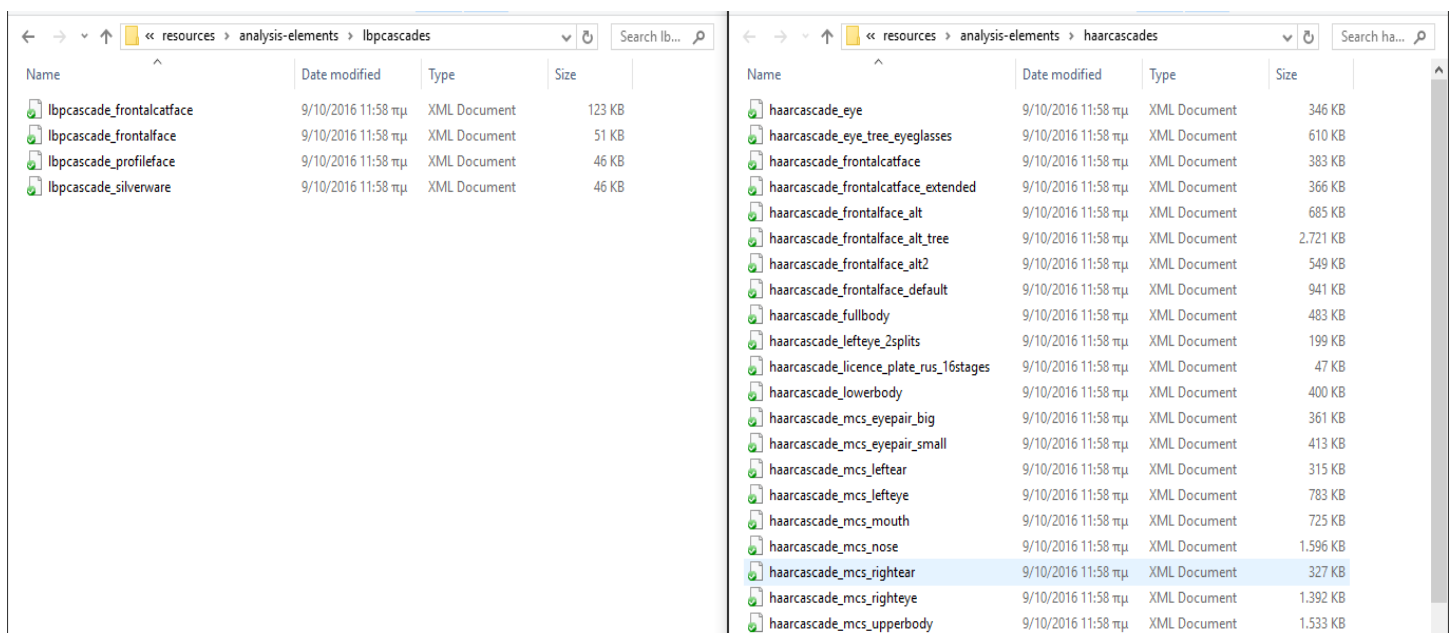


Figure 25 - XML Cascades - LBP on the left HAAR on the right

The previous figure shows the characteristics for each cascade category mention before. Both categories contain XML files for each characteristic. One of the difference is, the haar-cascades contain more characteristics. That helps for more precise detection of a face. But also make this category of cascades to be slower compare the LBP cascades and not only because the different amount of the characteristics. The Haar-cascades for every calculation, they are calculating, they are using floating numbers compared with the LBP cascades where they are using integer numbers for the calculations.

After the application detects the face or faces use the object pr type of PersonRecognizer. This object is responsible to gather the data for the training algorithm and after passing this data to the FaceRecognizer object fr which is provided from OpenCV and is the object responsible for the training/recognize the individual.

There are three ways to for fr object to be trained:

- Eigenface Recognizer - `createEigenFaceRecognizer()`
- Fisherface Recognizer - `createFisherFaceRecognizer()`
- Local Binary Patterns Histograms Face Recognizer - `createLBPHFaceRecognizer()`

The algorithms, Eigenface and Fisherface have two inputs. A threshold and number of components. The threshold applied in the prediction. If the distance to the nearest neighbor is larger than the threshold, this method returns -1. The number of components (num_components) is used to describe how many objects/ persons you want to recognize. If you leave this at the default (0) or set it to a value less-equal 0 or greater (c-1), it will be set to the correct number (c-1) automatically.

The algorithm LBPH needs five inputs. Radius, neighbors, grid_x, grid_y and threshold. Threshold works the same way as described in Eigenface and Fisherface method. The radius used for building the Circular Local Binary Pattern. The parameters, grid_x and grid_y, are the number of cells in the horizontal and vertical direction, 8 is a common value used in publications. The more cells, the finer the grid, the higher the dimensionality of the resulting feature vector.

After we select one from the previous algorithms, we begin the training process. The training process needs the images and the labels. Images are the photos, which user, gave in the beginning of the application, with importing photos or selecting them from the database. And labels are the name of the photos for example the first photo of the individual is 1 and the second is 2 and etc.

After the training process has finished the object fr is able to make a prediction. The prediction gives back a confidence number, that number is a percentage how sure that individual is the individual that algorithm tries to recognize. The confidence number is used, based on the result of the prediction, to draw a blue rectangle on the person face that had recognized.

Both training and prediction needs the images to be in grayscale format for all the algorithms mention before.

The threshold is 180 as default value recommended from the most sources. The 'confidence' value is actually the distance between the test and the closest found image.

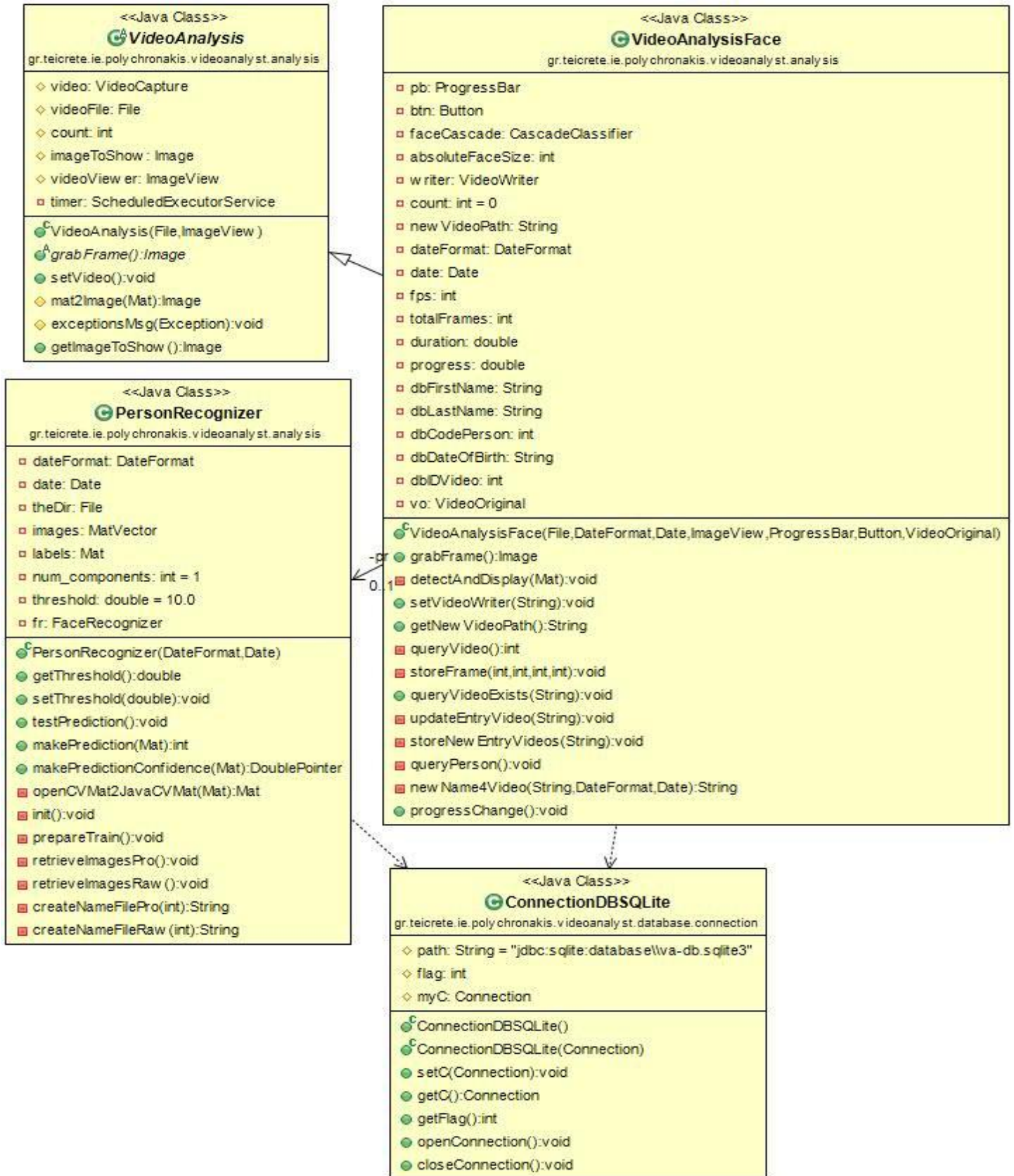
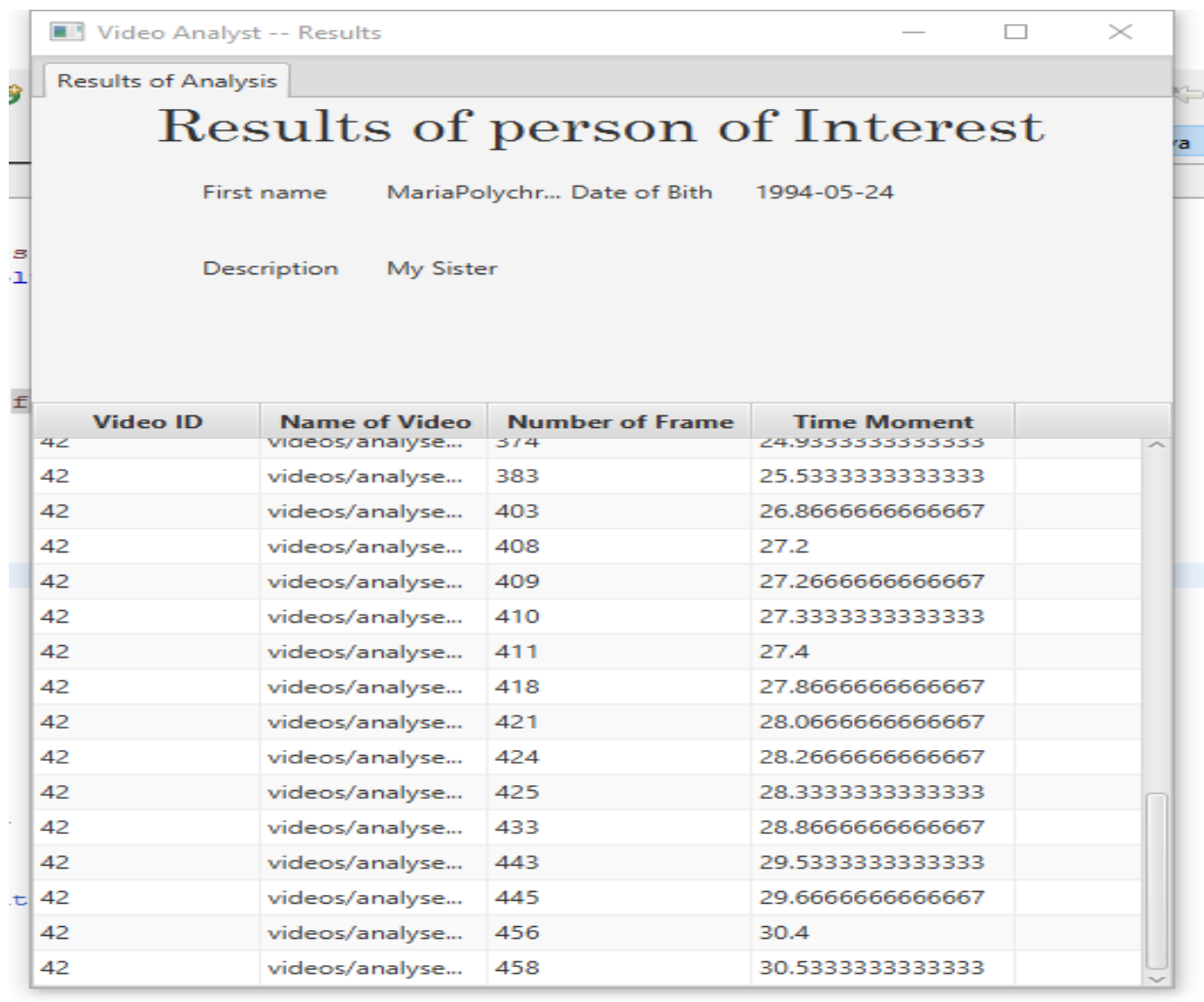


Figure 26 VideoAnalysis Class Diagram

3.2.5 RESULTS FRAME

After the analysis process has finished, the application returns the user back to PlayerFrame and simultaneously opens the ResultFrame.

The ResultFrame includes a small form with the data of the individual, which the application was analyzing the video for his/her presence, and a table view which his/her appearance was occurred. The table is filled from the database with the entries. The entries as mention before are from the process of the analysis. The table view shows the exact number of the Frame and the time moment where the individual was detected. Again to present the data in the table view we use the same technique was used in the table view of the “Import from Database” selection.



Video ID	Name of Video	Number of Frame	Time Moment
42	videos/analyse...	374	24.93333333333333
42	videos/analyse...	383	25.53333333333333
42	videos/analyse...	403	26.86666666666667
42	videos/analyse...	408	27.2
42	videos/analyse...	409	27.26666666666667
42	videos/analyse...	410	27.33333333333333
42	videos/analyse...	411	27.4
42	videos/analyse...	418	27.86666666666667
42	videos/analyse...	421	28.06666666666667
42	videos/analyse...	424	28.26666666666667
42	videos/analyse...	425	28.33333333333333
42	videos/analyse...	433	28.86666666666667
42	videos/analyse...	443	29.53333333333333
42	videos/analyse...	445	29.66666666666667
42	videos/analyse...	456	30.4
42	videos/analyse...	458	30.53333333333333

Figure 27 Result Window/Frame filled with Data

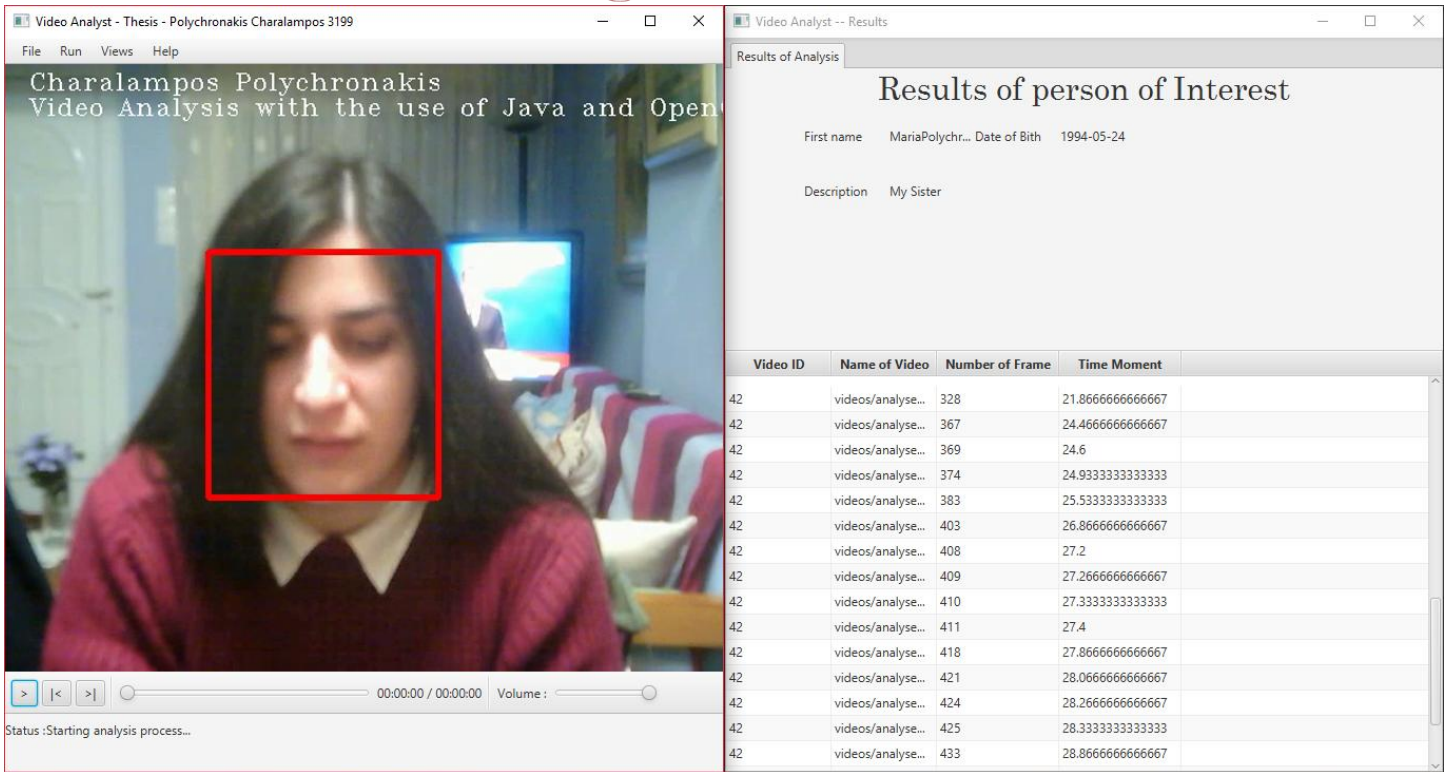


Figure 28 PlayerFrame and ResultFrame after the process of Analysis had finished

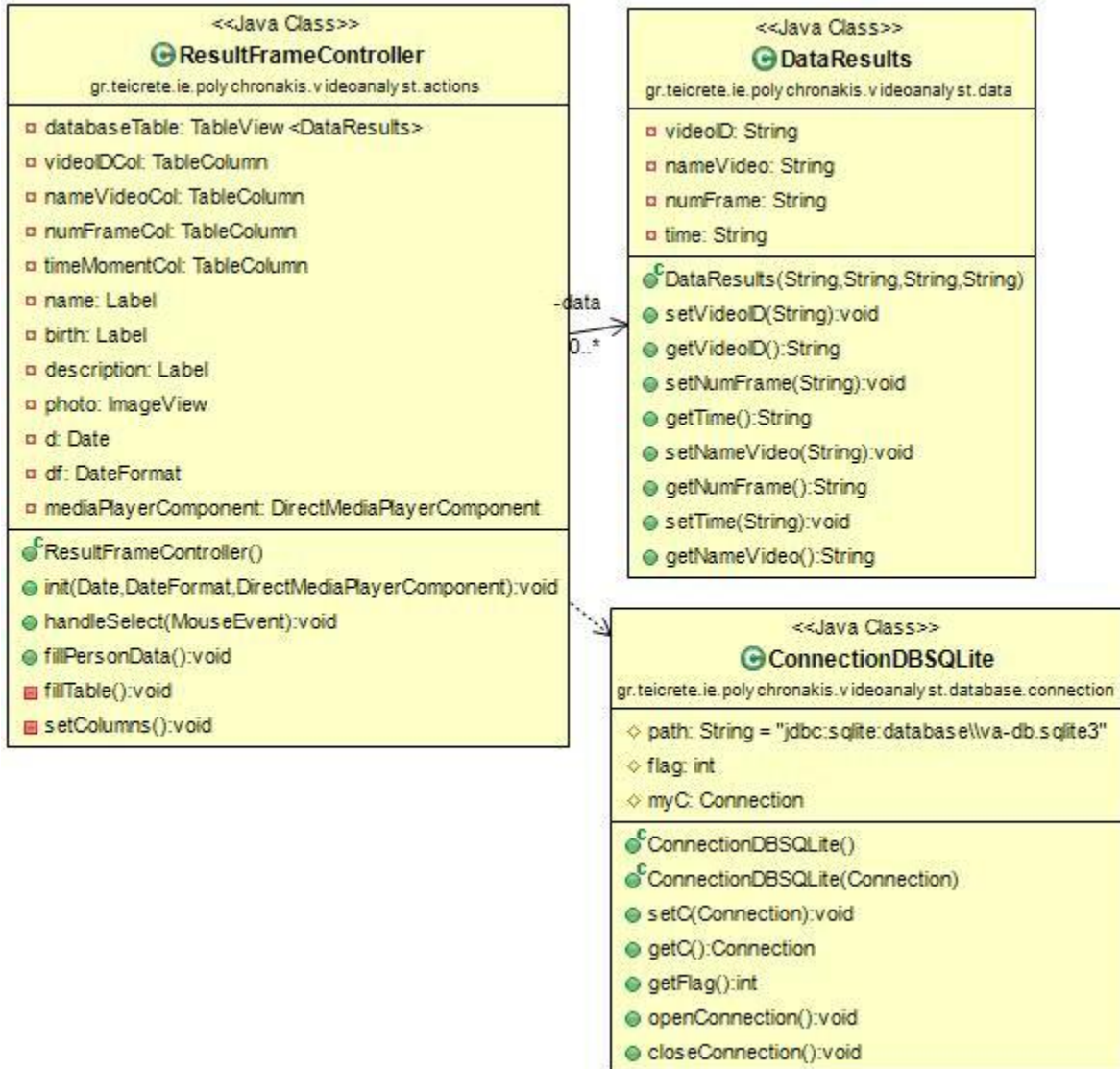


Figure 29 ResultFrame Class Diagram

3.3 CONNECTIVITY DATABASE

As it's mention, in the previous section many times, in the process to analyze the design of this application, there is a constant connection with a database. In the database, for the example of the application, it is stored information of the individual (va-persons), for the the videos (va-videos) and the connection between those two tables. For the individuals, they are stored the pictures that include them, before the image is processed (va-persons-raw) and after they pictures are processed. The class ConnectionDBSQLite is responsible for the connection with the database (Open,Close). The process to insert data to database happens to the classes the use database.

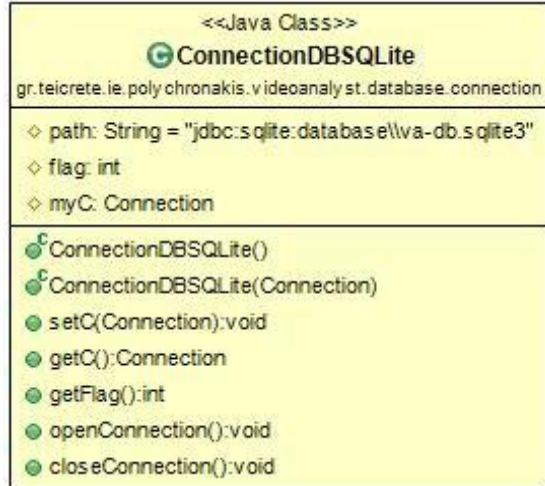


Figure 30 Connection Class - Class Diagram

In the next picture, the reader, can see the entity relationship of the current database, that been used by the application. In that point is good to mention, this database can change all the time and that depends how the application has been built and for what use and needs.



Figure 31 Entity and Relationship Model of the database

4 RESULTS

In this section, we would speak for the tests we had to run and their results.

The number of the individuals, that application, tried to recognize was 2 individuals. The first individual is Chari and the second is Maria. In the application, they are data for both from the perspective of the dataset, for the training, and also from the perspective of the video, for the recognition.

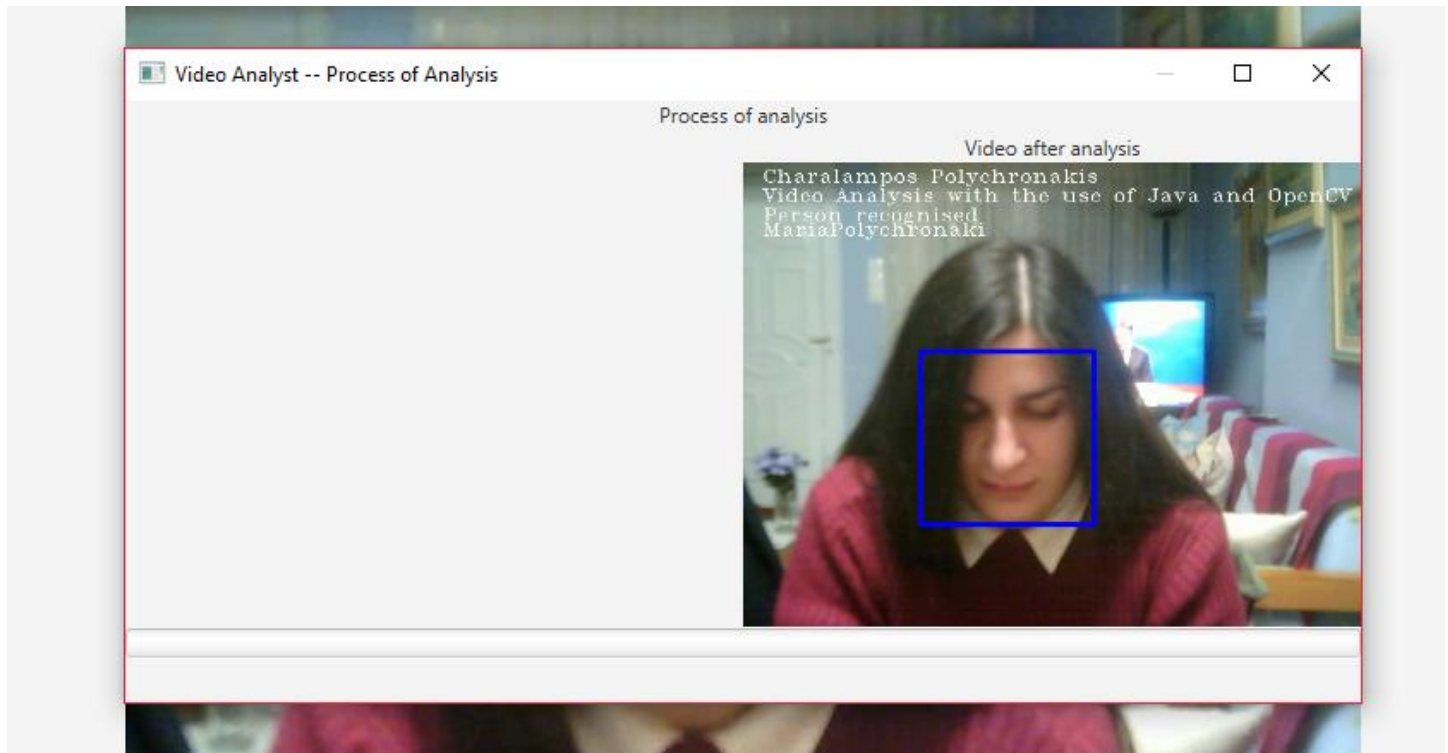


Figure 32 VideoAnalysisFrame during the analysis

The number of the experiments are three for each individual – one for each algorithm – totally 6. In the following sections will be comment only for the first individual, Chari, in three different section separate for each algorithm. In all experiments, the video input it was a video with both individuals on it and in the first 3 runs of the experiment the application was trying to identify the first individual, Chari, and the last 3 runs was trying to identify the second one, Maria. Of course is possible to try in a single run to identify both individuals at the same time, but that will be comment more in the next chapter also the fact why in this chapter why is not commented the circle experiments with the individual Maria.

The video that was used for recognized both individual, had the specific characteristics:

- durations: 00:00:31:9 (hh:mm:ss.sss)
- frame rate: 15 frames/second
- Frame width: 640 pixels

- Frame height: 480 pixels
- Total frames: $474 = (31\text{sec} * 15\text{frames}) + 9 \text{ FRAMES}$

Timecode used is in SMPTE format (HOURS:MINUTES:SECONDS:FRAMES) so the last number is FRAMES and not Miliseconds!

These characteristics is the same for each algorithm. So based on the number of frames, which are processed to see if the individual is on it, will be calculate the percentage appearance of the individual on the video.

Also, before the numbers, here are 4 examples what is wrong and what should be not calculate as a correct recognized frame.

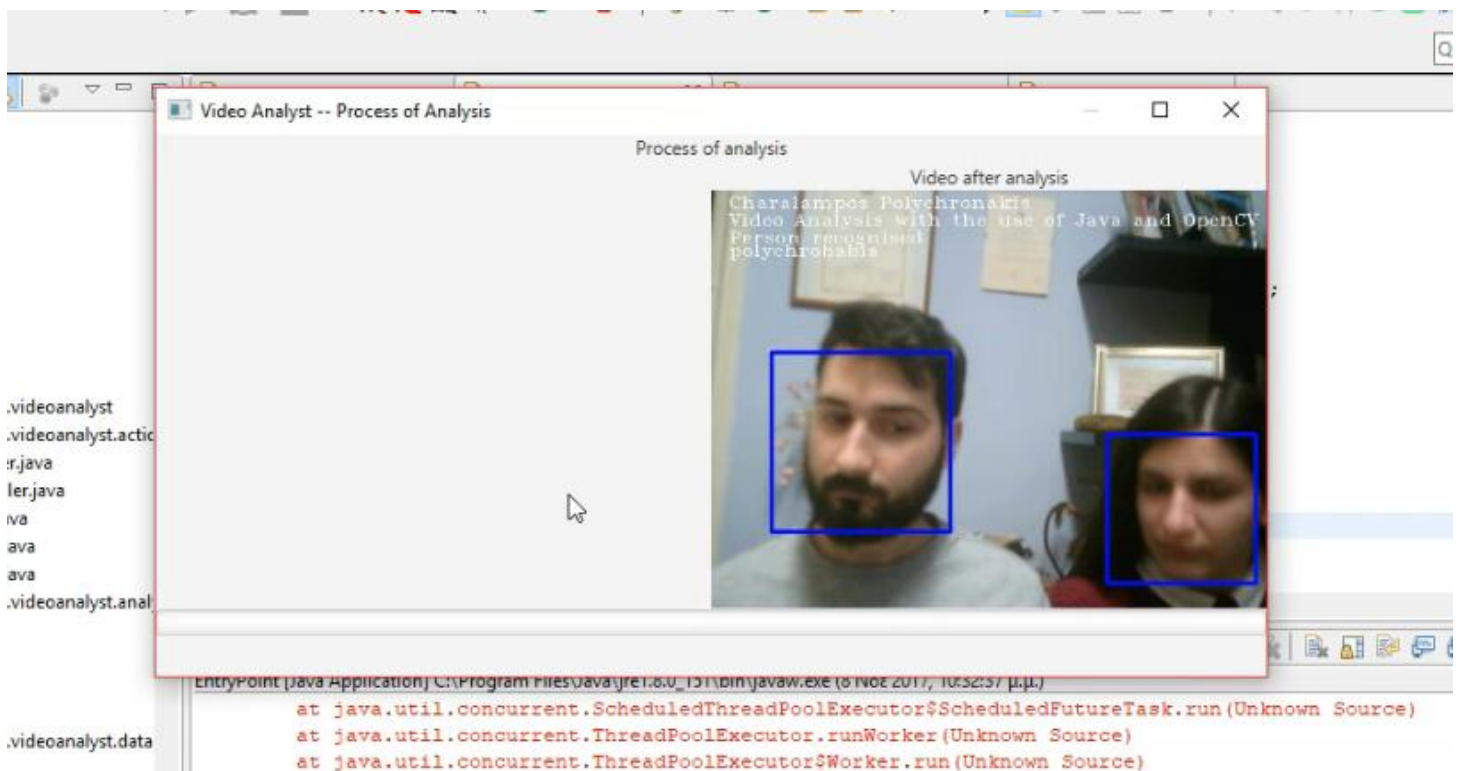


Figure 33 Wrong Recognition 1 - Both faces recognized as the same person

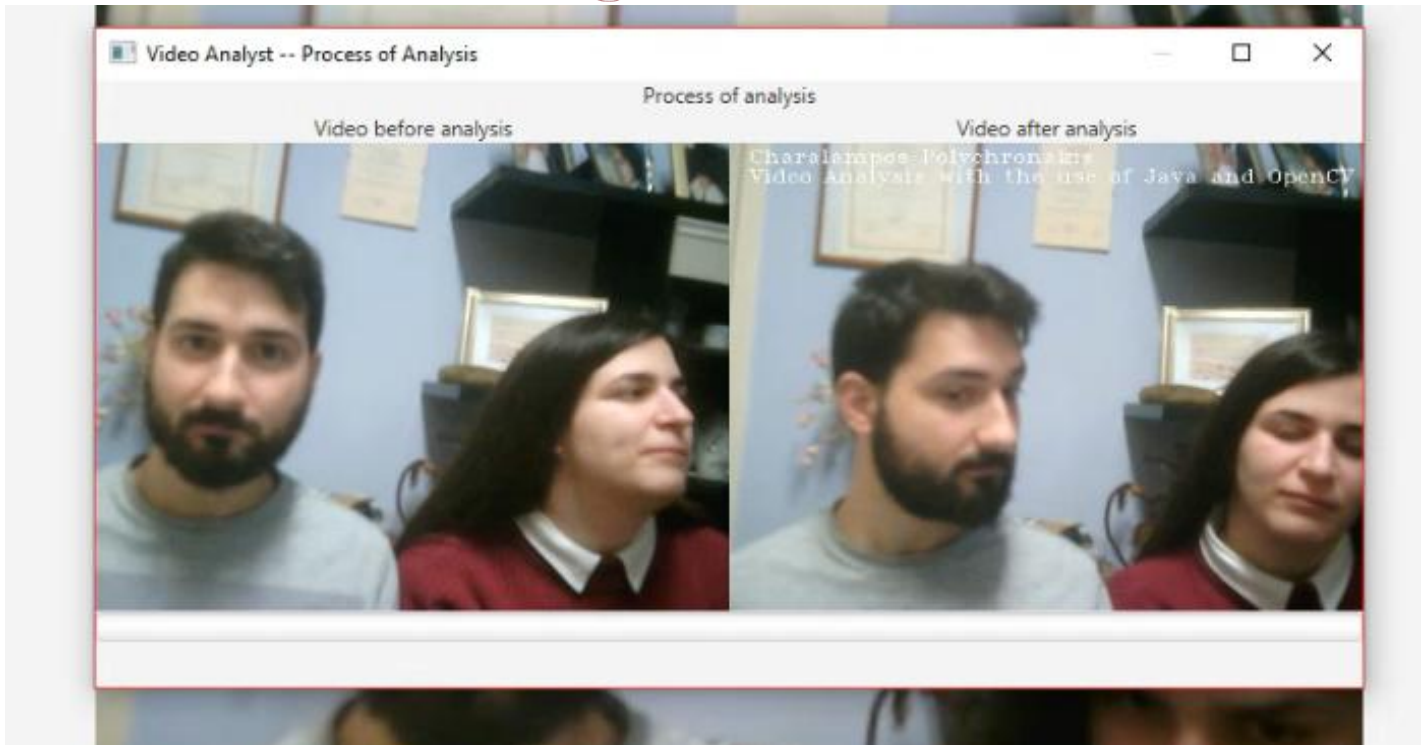


Figure 34 Wrong Recognition 2 – No face was detected

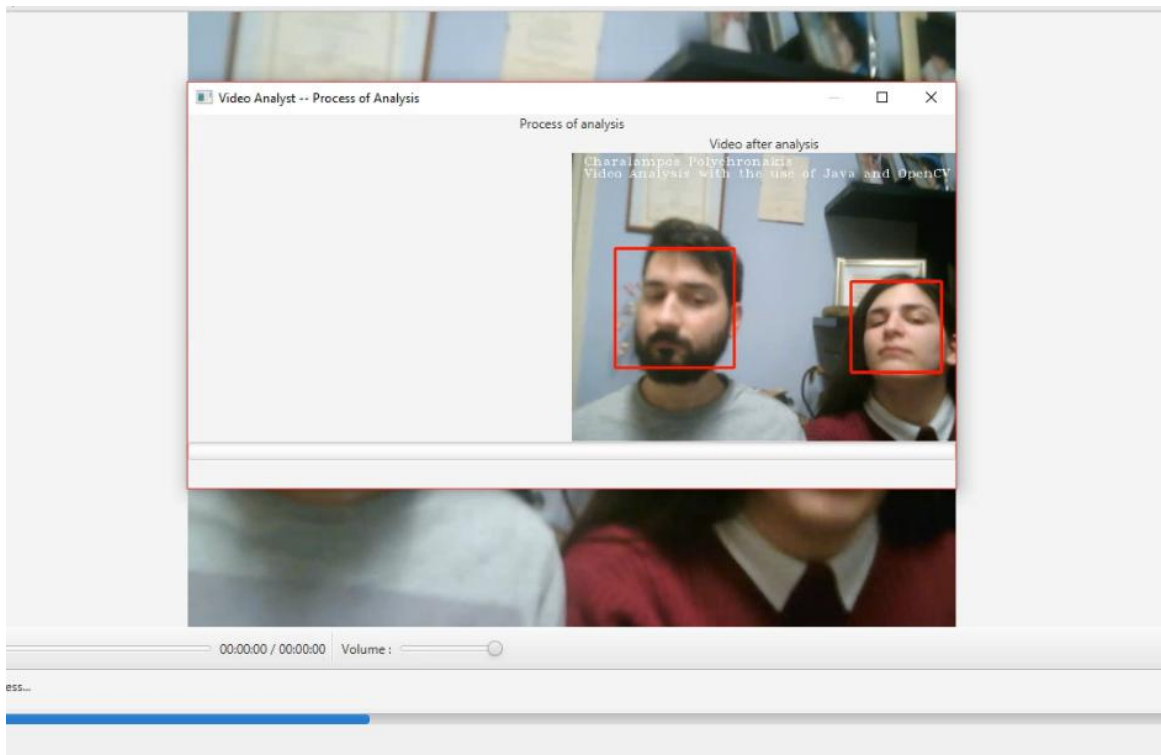


Figure 35 Wrong Recognition 3- No face recognized

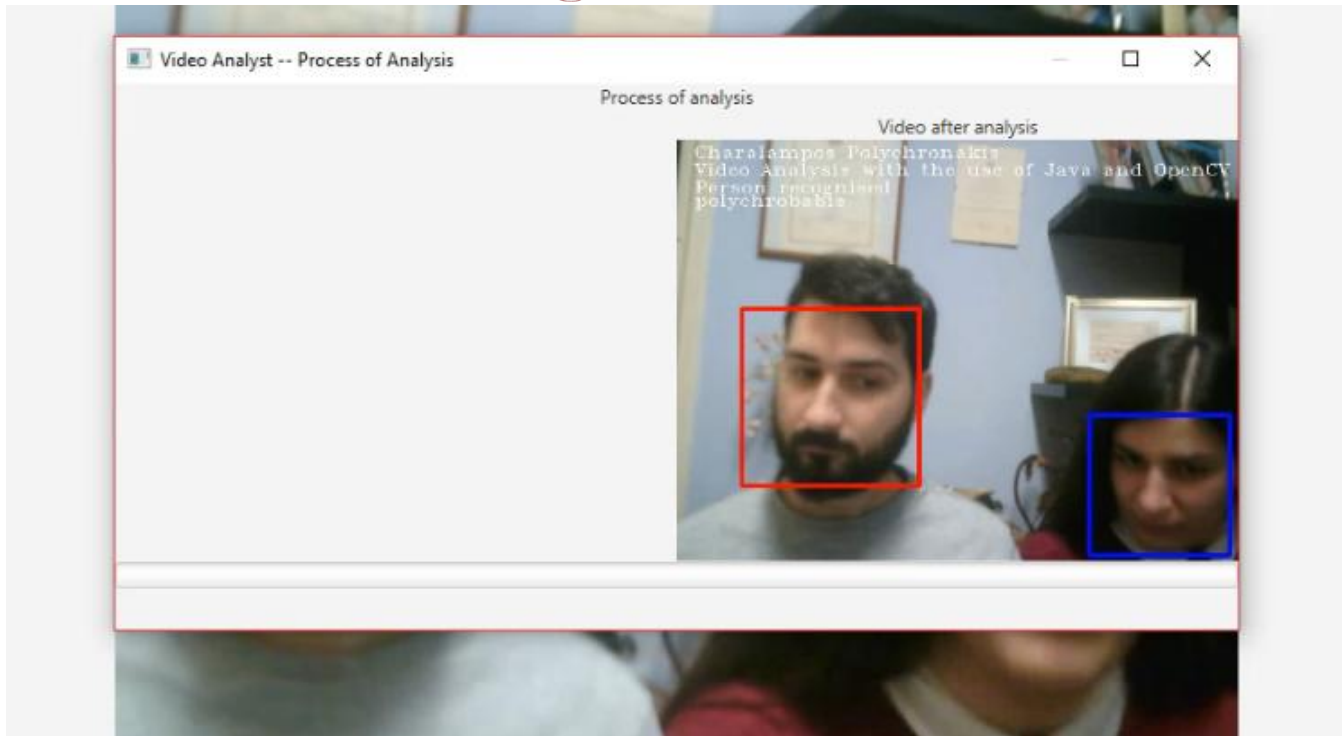


Figure 36 Wrong Recognition 4 - Wrong recognition of individual

4.1 RESULTS EIGENFACE

- i. In frames:
 - a. Number of the recognized frames: 59
 - b. Number of the correct recognized frames: 36
 - c. Number of bad recognition of the individual (not the proper one): 23
 - d. Number of rest frames (Missed recognition or not face): 415
- ii. In percentages:
 - a. Number of the recognized frames: 12.44%
 - b. Number of the correct recognized frames: 7.59%
 - c. Number of bad recognition of the individual (not the proper one): 4.85%
 - d. Number of rest frames (Missed recognition or not face): 87.55%

4.2 RESULTS FISHERFACE

- i. In frames:
 - a. Number of the recognized frames: 120
 - b. Number of the correct recognized frames: 87
 - c. Number of bad recognition of the individual (not the proper one): 33
 - d. Number of rest frames (Missed recognition or not face): 354
- ii. In percentages:

- a. Number of the recognized frames: 25.31%
- b. Number of the correct recognized frames: 18.35%
- c. Number of bad recognition of the individual (not the proper one): 6.96%
- d. Number of rest frames (Missed recognition or not face): 74.68%

Good to mention, compare with Eigenface algorithm, Fisherface algorithm has increase the correct recognized frames by 10.76% and also decrease the number of Missed frames by 12.87%

4.3 RESULTS LBPH

- i. In frames:
 - a. Number of the recognized frames: 266
 - b. Number of the correct recognized frames: 203
 - c. Number of bad recognition of the individual (not the proper one): 63
 - d. Number of rest frames (Missed recognition or not face): 208
- ii. In percentages:
 - a. Number of the recognized frames: 56.11%
 - b. Number of the correct recognized frames: 42.82%
 - c. Number of bad recognition of the individual (not the proper one): 13.29%
 - d. Number of rest frames (Missed recognition or not face): 43.88%

Compare with Eigenface algorithm, LBPH algorithm was able to increase the number of the recognized frames by 35.23% and also to decrease the number of the missed frames by 43.67%

And last comparison between Fisherface algorithm and LBPH algorithm, LBPH increase by 24.47% the recognized frames and decrease the missed frames 30.8%

5 CONCLUSION AND FUTURE WORK

In this final section of this thesis will be commented all the results and what is possible to change to improve these statistics, as a final conclusion. Also, there is going to be an idea for inspiration and for further work to improve this application and the possibilities of this application in real life.

5.1 CONCLUSION OF THE COMPARISON OF THE ALGORITHMS

From the previous chapter, based on the data, it is possible to understand the most accurate algorithm was LBPH.

But why is that happening? As has mentioned before in the theory, eigenface concerns more in characteristics with big differences (noses, mouth, eyes, etc.) from the training dataset, but with doing that you lose information. Fisherface as an improve of eigenface ,as has it mentioned before, with the difference from Eigenface, extracts principal components that differentiate one person from the others but not letting these components become dominant. Both algorithm use illumination changes as a basic future which is not accurate for extracting information, because not always it is possible to have the ideal lighting. To prevent that and have better recognition we use the LBPH algorithm. It doesn't use/cares the illumination changes, but tries to get the structure of the face just with comparing the neighborhood pixels.

That is the conclusion from the algorithmic side of that application, but it is not only that the issue we don't have an accurate recognition. As was mention in the previous chapter, we had one more individual, Maria. The data from this circle of tests for this individual is not presenting for the simplest reason that are not accurate. What that does it mean? Let's explain the first step, which is the training set.

From the follow figures (Figure 37, Figure 38), it is able to see the different dimension of an image for each individual. The image of Chari has bigger dimension than Maria. From that information is it able to perceive that in the training process, the training dataset for the individual Chari has more information how Chari looks like than Maria.

Let's consider that we have the ideal training dataset. Again there is the possibility not to have an accurate recognition. And that is because of the video. For the test, as mention before, the dimension of the video is 640 x 480. The application tries to predict if the individual is on the frame, and if, for example, the training dataset had trained with images with dimension like the ones of the individual Charis (Figure 38 – 1944x2592) and use to test a frame from the video mentioned before, again the application doesn't give a lot information to try to predict.

And last point, how much NOT precise can a video be, is on the frames of the video, if there is a person on it or not and the view of that person. The human brain can understand by the characteristics if there is face, but also can understand if the face is profile, up or down, a little bit rotate left or right and even some characteristics have been altered.

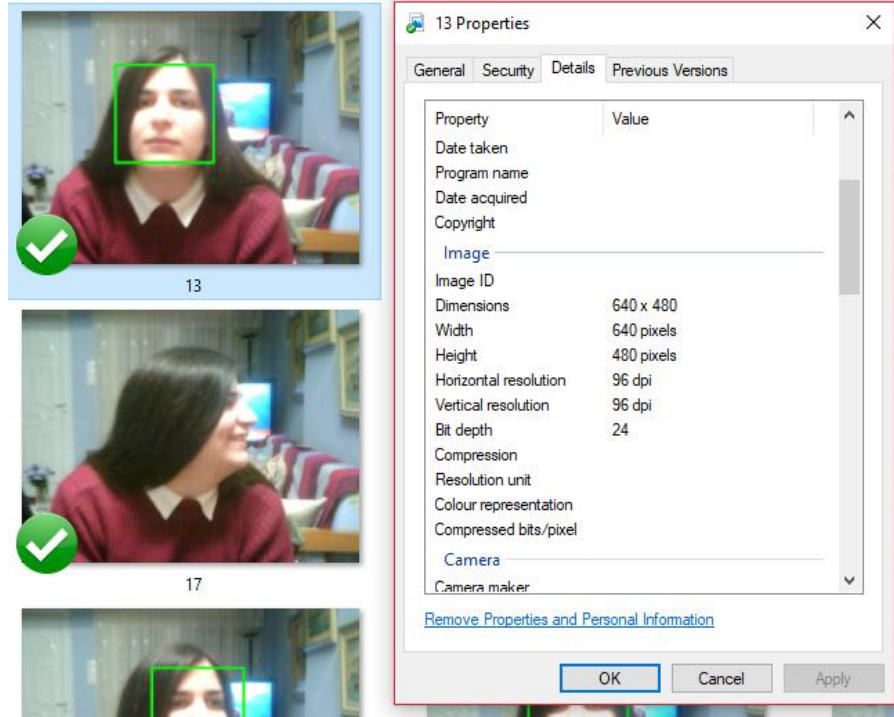


Figure 37 Image Details for the second Individual, Maria

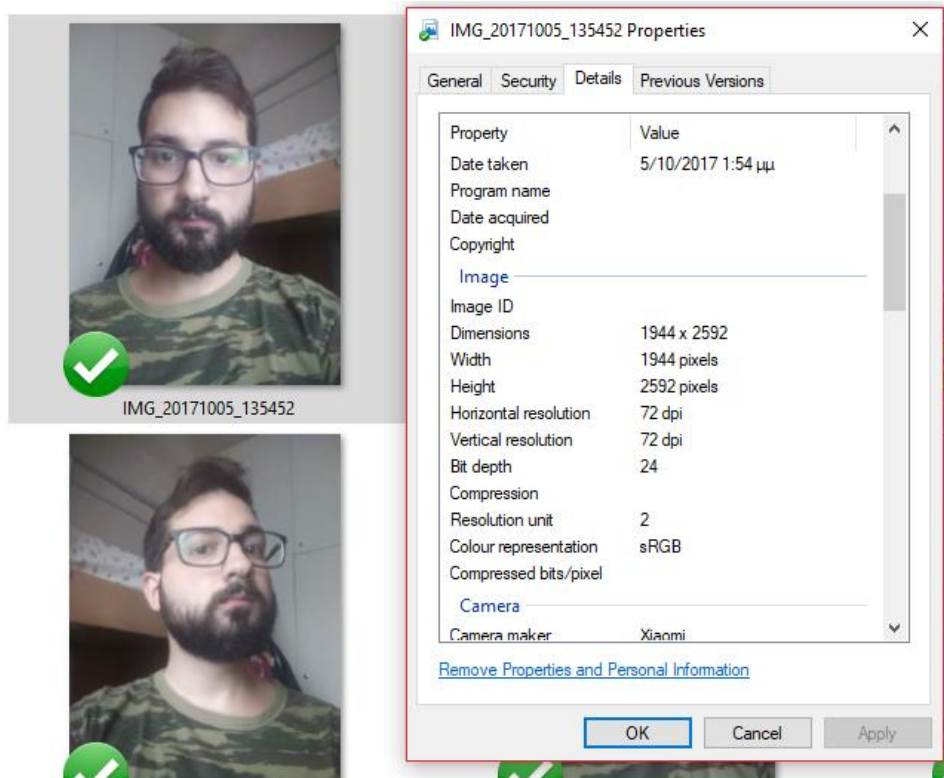


Figure 38 Image Details for the first Individual, Chari

5.2 FUTURE WORK

It was explained why the application is not accurate. What does it need to become more accurate?

In the last paragraph, it is mentioned that the view of the face may not be direct. There is way to implement more methods to try perceive from different perspectives/ views of the face of the individual. One simple example is if the face is down. Open CV gives that possibility to flip an image by just call the function `flip()`, and after that the application can continue the process of recognition. The same procedure if the face is rotated more to left or right side.

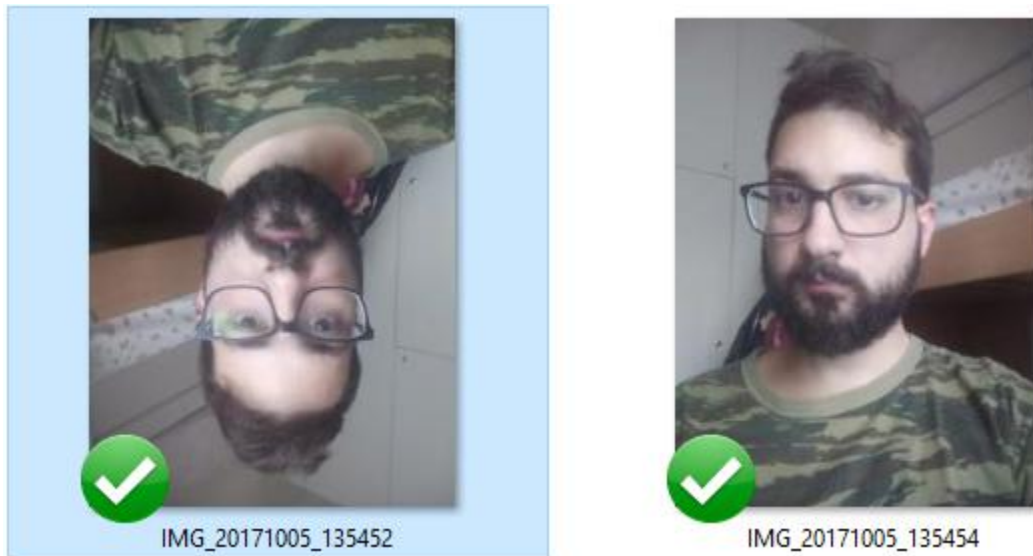


Figure 39 Example of UP & DOWN image

About the input of the data, it is good that the application should be in position to pre-analyze the data and pinpoint the user if they are good data for his/her purpose. And try to guide the user how to use the application. That part is sensitive

Also has it mentioned before, why not to try at the same time try to recognize two individuals at the same time? Again that is a simple process, anyone can try to create a second thread that tries for another individual to predict him/her. For simplicity that application tries to recognize one individual because the purpose of that thesis is to compare and pinpoint the problems of the algorithms.

Some extra addition for that application would be the prediction of emotion. There are a lot of open database that would be able to help any developer or engineer to create a training dataset that would be able to define in which frame is the individual and what is his/her status (happy, sad, frustrated, bored, surprised, etc.) Or even the application to be able to read a video and from its self to separate the individuals on this video.

Applications, similar to ours, that has to do with videos and now days predictions can be used for surveillance if you are looking for a terrorist or a criminal (in that point is good to mention, for these

purposes it depends the country and the laws about data surveillance), research for self-awareness with the use of Role Playing Games and try to pinpoint to the person that watches the video what was his/her actions and behaviour, research in medicine with recording a patient before and after the treatment (even if that is for psychology purposes or pathology purposes). [17] [18]

6 REFERENCES / BIBLIOGRAPHY

- [1] R. J.Radke, COMPUTER VISION FOR VISUAL EFFECTS, Cambridge University Press, 2013.
- [2] R. Davies, Computer and Machine Vision:Theory, Algorithms,Practicalities Fourth Edition, Academic Press, 2012.
- [3] Oracle, "What is Java?," 2017. [Online]. Available:
https://www.java.com/en/download/faq/whatis_java.xml.
- [4] OpenCV, "OpenCV - About," 2017. [Online]. Available:
<http://opencv.org/about.html>.
- [5] Oracle, "Java JDBC API," 1993. [Online]. Available:
<https://docs.oracle.com/javase/8/docs/technotes/guides/jdbc/>.
- [6] Oracle, "Java SE Technologies - Database," 2017. [Online]. Available:
<http://www.oracle.com/technetwork/java/javase/jdbc/index.html>. [Accessed 24 1 2017].
- [7] C. S. Limited, "vlcj," 2015. [Online]. Available:
<http://capricasoftware.co.uk/#/projects/vlcj>.
- [8] SQLite, "About SQLite," [Online]. Available:
<https://www.sqlite.org/about.html>.
- [9] S. Emami, "Introduction to Face Detection and Face Recognition," 2012 . [Online]. Available:
<http://www.shervinemami.info/faceRecognition.html>.
- [10] E. Arubas, "Face Detection and Recognition (Theory and Practice)," Saturday 6th April 2013.
[Online]. Available: <http://eyalarubas.com/face-detection-and-recognition.html>.

- [11] J. Howse, OpenCV Computer Vision with Python, 35 Livery Street Birmingham B3 2PB, UK: Packt Publishing Ltd., 2013.
- [12] O. Documentation, "Face Detection using Haar Cascades," 2017. [Online]. Available: http://docs.opencv.org/trunk/d7/d8b/tutorial_py_face_detection.html.
- [13] R. Raja, "FACE RECOGNITION USING OPENCV AND PYTHON: A BEGINNER'S GUIDE," 2017. [Online]. Available: <https://www.superdatascience.com/opencv-face-recognition/>.
- [14] OpenCV, "Face Recognition with OpenCV," 2017. [Online]. Available: https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#eigenfaces.
- [15] Oracle, "Class Application," 2008. [Online]. Available: <https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html>. [Accessed 2017].
- [16] O. J. Tutorials, "opencv-java-tutorials.readthedocs.io," [Online]. Available: opencv-java-tutorials.readthedocs.io/en/latest/o6-face-detection-and-tracking.html.
- [17] S. Rajaramnagar, "Implementation of Media Player using Image Processing and OpenCV JavaFx library," 2016. [Online]. Available: <http://ijesc.org/upload/2fb16de857ea72be2152aaf742d1de79.Implementation%20of%20Media%20Player%20using%20Image%20Processing%20and%20OpenCV%20JavaFx%20library.pdf>.
- [18] R. S. S. S. A. V. A. K. O. Barinova, "EFFICIENT ROAD MAPPING VIA INTERACTIVE IMAGE SEGMENTATION," 2009. [Online]. Available: <https://pdfs.semanticscholar.org/6368/c4d1320d45e4bed282a32c9973099e2ef2ee.pdf>.