Face Detection by Phantom 4 Pro+ Aircraft in Real Time

Roberto Contaldo, Francesco De Feo, and Sara Zaino

[1] University of Salerno
[2] Department of Informatics
[3] Master's Degree in Information Security
Project of Basics of Artificial Vision and Biometrics

Abstract. Nowadays, security is a top priority. In fact, biometrics uses cutting-edge technologies to identify terrorists and criminals. But the practice of distinguishing humans based on intrinsic physical or behavior traits goes back thousands of years.
There's evidence that fingerprints were used on clay tablets since Babylonian business transactions in 500 BC. Starting from this moment, other physical characteristics were widely used to distinguish people, specially between general public and criminals. Then, some police adopted the Bertillon system, invented in France, which recorded body measurements on index cards. However, errors were frequent because of the lack of unique standards. With the widespread use of computers in the late 20th century, new possibilities for digital biometrics emerged and new technologies were generously used.
Among these, we remember high resolution security video cameras and drones. So, the aim of the present project is to study and explain the features of these technologies, especially the ones of the the Phantom 4 Pro+ aircraft and analyze its operating methods in order to identify human faces during live streaming of videos. For this purpose, it will be used Paul Viola and Michael Jones' face detection algorithm, which includes Haar features and cascade classifiers to identify faces, eyes and ears of an individual.

Keywords: Face detection · Drone · Real Time · Phantom 4 Pro+.

1 Introduction

1.1 Motivations and Goals

Biometrics have been in use for thousands of years, thought not in the way we use them today. In fact, since prehistoric times, men used to sign cave paintings with their hands and feet; during the Old Age in Ancient Egypt, the construction of the Great Pyramid of Giza involved thousands of slaves who were paid thanks to anthropometric measurements. These are some of the first techniques employed in biometrics identification: hence, the importance of this kind of calculations, in order to guarantee security and recognition.

Nowadays, biometrics authentication is used in computer science as a form of access control and surveillance. It is way to identify and recognize people based on distinctive and measurable physiological characteristics that don't change over an individual's lifetime; they are related to the shape of the body, such as fingerprint, palm print, palm veins, hand geometry, retina, face or iris recognition. In the last few years there has been great progress in the automatic and instant verification of an individual's physical characteristics, especially by using new and original tools, such as drones and high-definition video cameras. To prove it, it's enough to think that just three years ago, Alan Butler, an attorney at the Electronic Privacy Information Center (EPIC) was not sure about the development drones could have in surveillance systems.
In this context, we have chosen to analyze the potentiality of aircrafts, especially the ones of the Phantom 4 Pro+: the aim of the present project has become, so, to study and test its functionalities. More in detail, our goal is the execution of a precise face detection during a live streaming recorded by the drone: to achieve that, we would build a complete system architecture, from the recording of the video to its investigation.

1.2 State of the art

The term ***biometrics*** is derived from the Greek words bio (life) and metros (to measure); it indicates the branch of learning dealing with body and behavioral measurements and calculations using statistical and mathematical methods. More in detail, it evaluates unique variables and that is why automatic biometric systems are more reliable than traditional procedures. They have only become available over the last few decades, due to significant advances in the field of computer processing. Many of these new automated techniques, however, are based on ideas that were originally conceived hundreds, even thousands of years ago.

One of the oldest and most basic examples of how humans take advantage of biometrics is face recognition. Since the beginning of civilization, they have used faces to identify known (familiar) and unknown (unfamiliar) individuals. This task became more complicated as populations increased and as more convenient methods of travel introduced many new individuals into small communities and, so, other body characteristics were used as a formal means of recognition.

It can be said that the earliest form of biometrics appeared thousands of centuries ago. In ancient caves, walls were adorned with paintings created by prehistoric men who left numerous hand prints as a signature; also Babylonian business transactions were recorded in clay tablets that included fingerprints. Even in China, early merchants used them to settle business transactions, while parents used both fingerprints and footprints to discriminate a child from one another.

In early Egyptian history, traders were identified by their physical descriptors to differentiate between trusted traders of known reputation and those new to the market; also slaves were recognized by height and length of their arms in order to receive a correct salary payment.

But it was in the mid-1800s, that a more formal way to recognize and identify people became necessary, because of the rapid growth of cities due to the industrial revolution. Merchants and authorities faced with increasingly numerous populations, so they could no longer rely just on their local knowledge of individuals. This situation produced the need of a formal system to record criminals with their measured physical traits. So, in 1870, Alphonse Bertillon, a French anthropologist, developed a system of measuring various body dimensions, used for the first time in the Parisian prisons, that became known as *Bertillonage*. His method consisted on precise anthropometric measurements (such as height and width of arms and fingers, distance between elbows and end of the fingers, length of trunks and feet) calculated on prisoners' body because each skeleton is different from another one and the human backbone doesn't change beyond the twentieth year of age.

Then, in 1960 in USA the first real recognition system was developed and, in 1999, NSA (National Security Agency) installed many biometrics emplacements to discern among fingerprints, voices and faces; in favor of this, there was the idea that external support could be lost, stolen or left at home, while access keys can be forgotten, copied or revealed to someone else.

From this moment, lots of things changed and lots of advances were done: technology improved, and biometrics began to be used for many services and in various activities including banking, politics or research of "wanted" individuals. Nowadays, biometrics is often used in daily life as a fingerprint recognition feature that allows users to unlock their devices.

For a full comprehension, we can divide biometrics features into two groups (see Fig. 1):

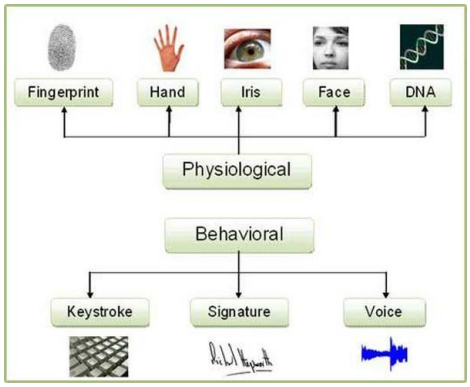

Fig. 1. Most famous and widespread biometrics systems

– *physiological*: these are related to human measurements, such as fingerprints, faces, palms, retina, iris, DNA, ear's shape, height or weight;

– *behavioural*: these are related to one's behaviour, such as handwriting, walk, keystroke, signature or voice.

Generally, the first ones are more reliable than the others, because individual's body shapes go through just a few changes during his whole life, while behavioural features are affected by psychological conditions, so they need to be changed frequently.

1.3 Drones

As said previously, among the most famous and widespread biometrics technologies we can find face recognition, that has recently gained success as a valid application for the analysis and the comprehension of images.

In the past, facial recognition software has relied on a 2D image to compare or identify another 2D image from the database, but a newly-emerging trend in facial recognition software uses a 3D model, which claims to provide more accuracy. They can also be used in real time to easily and quickly recognize who is the person located in front of the sensor (camera or webcam). Today, software for facial recognition deals with artificial intelligence algorithms, which make identification process automatic and almost immediate; this allows recognition or verification of one's identity just thanks to one or more images.

Face detection can be done using both fixed framing cameras, which can record just what appears in front of them, so that large areas cannot be totally under their control, and auto-framing cameras, which make it is possible for the frame to move in relation to users' movements.

At the moment, devices known as *drones* are already on the market; they are provided with professional and traditional cameras and can move in the surrounding space independently or in a remote-controlled way. Thus, lots of agencies supposed to use them for biometrics recognition systems, taking advantage of cameras' video signals.

Drones, as known as unmanned aerial vehicles (**UAV**), are aircrafts which can perform autonomous pilot. They can easily reach locations which are too difficult to reach or dangerous for human beings and collect images from bird's-eye view through aerial photography. They are made from different light composite materials in order to increase maneuverability while flying and reduce weight and can be provided with a variety of additional equipment, including cameras, GPS guided missiles, Global Positioning Systems (GPS), navigation systems, sensors, and so on.

UAV manufacturers often build in specific autonomous operations, such as *return-to-home*, that enables drone to fly back to the point of take off (often gaining altitude first to avoid possible intervening obstructions such as trees or buildings or *follow-me*, which maintain relative position to a moving pilot or other object, image recognition or homing beacon. So, drones are used in many areas and there is no end when it comes to their possibilities; one of them is identify people on the ground which is important for a variety of applications, such as surveillance, people search, and remote monitoring.

Since drones take pictures from the air, their altitudes generate angles of depression between drones and their targets that influence the poses of the faces in the collected pictures; even the speed of the flight can change the quality of the result. UAVs could be used to solve this problem since it could be assumed to automatically orient the camera in an optimal position, adapting frames continuously; in this case you cannot think to delegate this control to an operator because it would be slow and imprecise.

1.4 DJI Phantom 4 PRO+

DJI is the world's largest producers of commercial drones: all the drones produced by this brand receive positive feedback from the users, but the one that beats all other models in its category is the new Phantom 4 Pro+ drone.

This model takes care of every issue a flyer could come across. With an interesting camera and a long flight time, this drone is a treat for all professional photographers, hobbyists, and flyers. It comes with a system that avoids obstacles and it has a 4K camera and lens with an amazing sharpness, so videos and photos have the best quality possible. Besides, it has an advanced 3 axis gimbal that makes sure that there are no unnecessary vibrations and in-flight movements that might actually compromise the camera's capabilities.

Phantom 4 Pro+ (fig.2) is made with a magnesium framework, one that gives it the required strength while keeping it lightweight. This is one of the best features of the device as a lightweight drone is always expected to run longer. Add to that, the components are kept stiff and nicely attached to one another such that the drone encounters minimum or no vibration while flying in the air.

Fig. 2. Drone Phantom 4 Pro+ Pro

The controller has been made really comfortable for the DJI Phantom 4+; the news in this version is that it comes with its own smartphone. This makes things very convenient for the users, even if this has been one of the main problem we encountered during the development of our project; you would require a USB cable to make the connection between the phone and the controller.

1.5 Related Studies

The realization of a biometric system is especially influenced by benefits that derive from implementation and overall cost, which include sensors and related infrastructures. We can define a biometric system as a technological system that uses information about a person (or other biological organism) to identify that person. A complete and good one should be easy to use for any user, well-received by population and sufficiently resistant in case of fraudulent attacks: in fact, a biometric system could be attacked in order to compromise information security (data integrity and availability). It involves running data through algorithms for a particular result, usually related to a positive identification of a user or other individual. Specifically, users enter an account, user name, or inserts a token such as a smart card, but instead of entering a password, a simple touch with a finger or a glance at a camera is enough to try to authenticate them. Then, the system had to decide whether a person is really who he declares to be: the authentication system verify the identity by searching in a database for a match based solely on the biometric.

This process of recognition could be static, if it happens using a single image, or dynamic, if face acquisition takes place through a series of frames: in the first case images are normally obtained observing one's posture, illumination and background, producing high quality photos. In the second case, instead, there are more frames extracted by the video but their quality is lower due to irregular backgrounds and postures.

Many researches have explained and examined in depth how algorithms of face detection and recognition work. Among these, one that turned out to be useful is the "Rapid Object Detection using a Boosted Cascade of Simple Features", written by Paul Viola and Michael Jones, the creators of the famous algorithm of the same name. In it, they presented Haar feature-based cascade classifiers, that is an effective object detection method. More in detail, it is a machine learning based approach where a cascade function is trained from a lot of positive and negative images.

Part of their trick was to ignore the much more difficult problem of face recognition and concentrate only on detection, as we also did. They also focused only on faces viewed from the front, ignoring any seen from an angle. Given these bounds, they realized that the bridge of the nose usually formed a vertical line that was brighter than the eye sockets nearby. They also noticed that eyes were often in shadow and so formed a darker horizontal band. So, Viola and Jones built an algorithm that looks first for vertical bright bands in an image that might be noses; then looks for horizontal dark bands that might be eyes and, at last, it looks for other general patterns associated with faces.

The two same authors wrote another important paper,"Robust Real-Time Face Detection", which describes a face detection framework that is capable of processing images extremely rapidly while achieving high detection rate. A basic implementation of it it's included in OpenCV.

2 Methodology of the Study

Facial recognition technology is considered part of biometrics: lots of applications use software to identify or verify a person by mapping facial features, characteristics, and dimensions and comparing that informations with immense databases of faces.

These systems use a huge number of measurements and technologies to scan faces, including thermal imaging, 3D face mapping, cataloging unique features (also called landmarks), analyzing geometric proportions of facial features, mapping distance between key facial features, and skin surface texture analysis.

Specifically, we can say that landmarks are the different peaks and valleys that make up facial features. They are numerous, distinguishable and, in fact, each human face has approximately 70 of them. Some of these are: distance between the eyes, width of the nose, depth of the eye sockets, shape of the cheekbones, length of the jaw line.

Face detection can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of all objects in an image that belong to a given class; face detection algorithms focus on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit; in this case, an image matches with the one stores in database. Any facial feature changes in the database will invalidate the matching process.

2.1 How it works

Face detection technology might begin by searching for human eyes. It might do this by testing valley regions in the *gray-level image*. Then the genetic algorithm is used to generate all the possible face regions which include eyebrows, irises, nostrils and mouth corners.

Each possible face candidate is normalized to reduce both the lightning effect, which is caused by uneven illumination, and the shirring effect, which is due to head movement. The fitness value of each candidate is measured based on its projection on the eigen-faces. This name indicates a set of eigenvectors of the covariance matrix of the set of face images, where an image with N pixels is considered a point (or vector) in N-dimensional space. After a number of iterations, all the face candidates with a high fitness value are selected for further verification. At this stage, the face symmetry is measured and the existence of the different facial features is verified for each face candidate.

2.2 Applications of face detection

Face detection can be used for a lot of purposes. Some of them are:

Facial Recognition - A facial recognition system matches an individual's face instantly against a database of photographs in order to establish identity. This kind of software is used in a variety of ways, but most often for security

and law enforcement purposes, for example in airports to scan faces of travelers to search for individuals suspected of a crime or terrorists.

Law enforcement - Law enforcement uses facial recognition software to identify and apprehend people who commit crimes, but several states use facial recognition software also to prevent people from getting fake identification cards or driver's licenses.

Photography - It serves as a way to help cameras autofocus on peoples' faces and recognize smile.

People Counting and Marketing - Face detection is being used by some marketers in order to detect when people walk by a certain area and with its algorithms they can predict age, gender and other factors in order to serve up relevant advertisements.

2.3 Limitations

One of the strongest positive aspects of facial recognition is that it is non-intrusive. Verification or identification can be accomplished from two feet away or more, and without requiring the user to wait for long periods of time or do anything more than look at the camera.

However, there are limitations about image acquisition:

- Poor resolution images and poor lighting can reduce the accuracy of face-scanning results.
- Different angles and facial expressions, even a simple smile, can pose challenges for face matching systems.
- Facial recognition loses accuracy when the person is wearing items like glasses, hats, scarves, or hair styles that cover part of the face. Makeup and facial hair can also pose issues for face detection programs.
- Facial scans don't necessarily connect with a profile, meaning that a scan of a person's face may not be useful if there are no photos of them in an accessible database. Without a match, the identity of the person behind the face scan can remain a mystery.

2.4 Viola-Jones object detection framework

The ViolaJones object detection framework is the first object detection framework to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones [1, 2]. It can be trained to detect a variety of object classes, but it is generally used to solve the problem of face detection, as we also did during the development of this project.

The basic principle of this algorithm is to scan a sub-window capable of detecting faces across a given input image; although human can do this easily, a computer needs precise instructions and constraints. In order to obtain the expected outcomes, ViolaJones requires full view frontal upright faces pointing towards the camera. It might seems that these constraints could reduce the algorithm's utility somehow, but it is due to the fact that the detection step is

often followed by a recognition phase, so, actually, these limits on pose are quite acceptable.

The main characteristics it needs to be considered a good detection algorithm are:

- **Robust**: it must rate frames quite successfully.
- **Real time**: for practical applications at least two frames per second must be processed.
- **Face detection**: the goal is to distinguish faces from non-faces.

First of all, we should specify that all human faces share some similar properties. For example, the eye region is darker than the upper-cheeks or the nose bridge region is brighter than the eyes. So, when you adopt this algorithm, you should look at the position and size of the eyes, the mouth and the bridge of nose and at the oriented gradients of the pixels intensities. These regularities may be matched using **Haar Features** (see Fig. 3), which involve the sums of image pixels within rectangular areas. It is a machine learning based approach created by Paul Viola and Michael Jones where a cascade function is trained from a lot of positive (face) and negative (non-face) images and it is used to detect objects in other images, by training on hundreds of thousands of face and non-face images to learn how to classify a new one correctly. So, we basically we call it *classifier*.

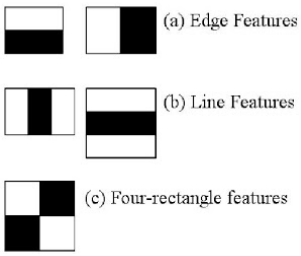

Fig. 3. Facial landmarks points

However, the algorithm has four stages: selection of the just mentioned Haar features, creation of an integral image, training using Adaboost [3] and then organize them in something called a classifier cascade.

1. **Haar Feature Selection** - Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier, because we need to extract features from it. Each feature is a single value obtained by subtracting the sum of pixels under white rectangle from the sum of pixels under black rectangle.
2. **Creation of an Integral Image** - All possible sizes and locations of each picture are used to calculate plenty of features. For each feature calculation,

we need to find the sum of pixels under white and black rectangles. To solve this, they introduced the integral images: it simplifies calculation of sum of pixels, how large may be the number of them, making it an operation involving just four pixels.

3. **Adaboost Training** - Among all these calculated features, most of them are irrelevant. A fast way to select the best ones out of 160000 (which are contained in a simple 24x24 window) or more is achieved by Adaboost [4, 5]; it selects nearly 6000 images and continues to work on them. In order to do it, we apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative; obviously, there would be errors or misclassifications. So, We just select the features with minimum error rate, which means they are the features that best classifies the face and non-face images. Final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. According to Viola and Jones, even 200 features provide detection with 95% accuracy.
4. **Cascading Classifiers** - In a photo, most of the image region is non-face region. So it is a better idea to have a simple method to check if a window is not a face region. If it is not, the algorithm discard it and doesn't process it again. Instead, it focuses on regions where there can be a face. This way, we can find more time to check a possible face region. For this reason, they introduced the concept of Cascade of Classifiers. Instead of applying all the 6000 features on a window, it groups features into different stages of classifiers and applies one-by-one (normally first few stages will contain very less number of features). If a window fails the first stage, it is discarded and the remaining features on it aren't considered. Instead, if it passes, the algorithm applies the second stage of features and continue the process. The window which passes all stages is a face region.

3 Experiments and Results

3.1 Analysis of the problem

At the beginning, our goal was to detect faces and eyes in real time using a live video streaming produced by a Phantom 4 Pro+. In fact, with the spread of new technologies, such as drones, there have been a lot of innovations and new services have been offered in many fields like law enforcement, photography or, obviously, security.

Hence, the required creation of a support in the identification of human features that could happen in **real time**, taking advantage of high quality devices and high resolution cameras.

So, to achieve the aim of our project, we tried many softwares, applications and libraries: some of these gave us problems or seemed to slow down the visualization of the streaming, so we looked for the best free compromises we could find searching on the internet.

What we produced at the end of these studies was a real *system architecture* (see Fig. 4).

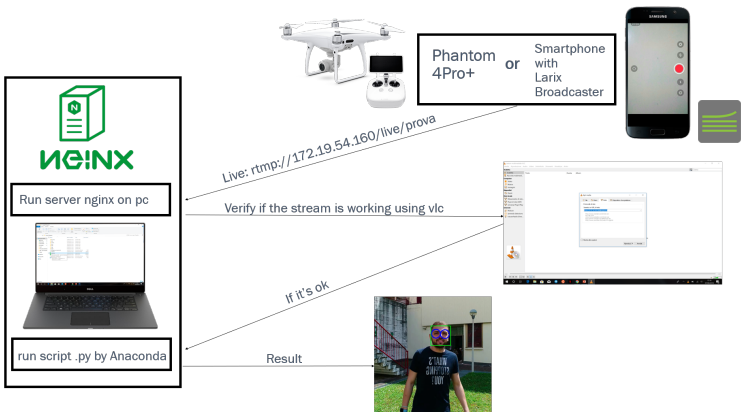

Fig. 4. System architecture

3.2 Problems faced

How to get the video of live streaming First of all, we have to find a way to get the video of the live streaming filmed by the Phantom in real time, because face detection has to happen at the same time as the drone is recording; so, you can simultaneously see the video with eyes or faces identification and the one without it.

A system that could help us for this purpose was a server: on account of this, we started to search for one that was compatible both with the tools available on the aircraft and the one at our service. We found out that DJIGO4 App can transmit in several platforms, including YouTube, Facebook, Weibo and in a customized one.

Hence, our second choice was the latter option offered by the app: it allows you to transmit using **RTMP** (Real-Time Messaging Protocol), a TCP-based protocol which maintains persistent connections and allows low-latency communication. It was initially a proprietary protocol developed by Macromedia for streaming audio, video and data over the Internet, between a Flash player and a server.

The "plain" protocol uses TCP port number 1935 by default. To deliver streams smoothly and transmit as much information as possible, it splits streams into fragments, and their size is negotiated dynamically between the client and server. Sometimes, it is kept unchanged; the default fragment sizes are 64 bytes for audio data, and 128 bytes for video data and most other data types. Fragments from different streams may then be interleaved, and multiplexed over a

single connection. The RTMP defines several virtual channels on which packets may be sent and received, and which operate independently of each other.

One of the most important advantages of RTMP, when it comes to live broadcasts, is the very low delay from real time in transmission; HTTP based technologies, instead, introduce very high delays and that just cannot work with live transmissions such as sport events or betting systems.

Anyway, the problem turned into finding a server able which allowed us to keep the same protocol we used during the streaming, considering that we cannot change it. Among all, we chose **nginx**, which seemed able to transmit the streaming very quickly and to avoid losing even a single frame.

It is a web server which can also be used as a reverse proxy, load balancer, mail proxy and HTTP cache; it can also be deployed to serve dynamic HTTP content on the network and it can serve as a software load balancer.

Nginx uses an asynchronous event-driven approach to handling requests. Its modular event-driven architecture can provide more predictable performance under high loads. The default configuration file is named `nginx.conf`.

It was exactly this file that gave us many problems during this step. In fact, we had to change some parts of it and add others to allow a stream transmission using RTMP protocol (see Fig. 5).

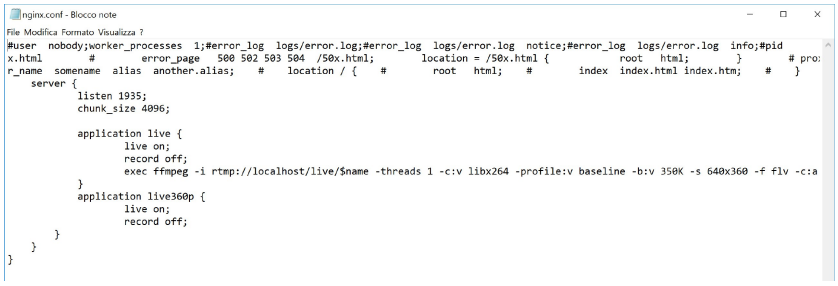

Fig. 5. Configuration file

We attached the code lines that specify that the listening port of the server is 1935 (which is the default one of RTMP) and the setting needed to record a live video. In order to do it, we use a software called **ffmpeg**: it provides a command-line tool that converts audio or video formats and it can also capture and encode in real-time from various hardware and software sources.

You need to disable firewalls or create a rule for inbound connections to port 1935 to start accurately nginx, ideally running it as an administrator.

How to process video Got the video, we had to process it in order to apply face detection algorithms. Initially, we searched for way to link RTMP protocol with **MATLAB**, a multi-paradigm numerical computing environment and pro-

prietary programming language developed by MathWorks, that we would like to use for the implementation of the biometrics calculations.

However, we found out that nothing similar to what we were imagining existed; but, during our tests, `ipcam` objects come out: they are used to acquire one image frame from an IP camera and display it. Unfortunately, they only works with HTTP or RTSP protocols, so this function cannot be compatible with our data flows. For this reason, we decided to avoid the problem converting the format of the streaming from RTMP to RTSP. Anyway, we didn't found any software that could make this conversion free.

Since we were using VLC to test if the streaming was successful, we thought to take advantage of it also to create a connection with MATLAB; again, there was no way to do it.

3.3 Solutions

The only solution that could be useful to realize the processing of the live streaming was Python: so, we started using it and some of its libraries in order to write code about Viola Jones' algorithm.

Librtmp One of the first one we analysed was librtmp: it is a RTMP client library and uses the implementation provided by librtmp via cffi.

However, it gave us many problems because of the operating system we were using to process the video streaming; therefore, we decided to change it.

OpenCV Among all the others, OpenCV library appeared as the best solution for our goal. It is a set of programming functions mainly aimed at real-time computer vision, built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. In fact, it was designed for computational efficiency and with a strong focus on real-time applications.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and machine learning ones. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality.

Lot of well-established companies make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

Final implementations The last problem we had to face was the choice of the operating system that could allow us to detect faces from the live as fast as possible.

First of all, we used a **Linux virtual machine**, selecting its version 18; then, we installed Python 2.7 and OpenCV library. With this architecture, we managed part of the problem; however, we couldn't fully take advantage of the potentiality and performances of the computer we were using to test.

This brought us to move our architecture in Windows, in order to use the face identification script in **Anaconda**, a free and open source distribution of the Python programming languages for data science and machine learning related applications, aimed to simplify package management and deployment.

It seemed to us as the best solution because it makes you able to create an environment where you can already have Python from the first execution of Anaconda and you can install all the other libraries you need, as we did with OpenCV.

The change of the operating system has been done not only to allow the execution of the Python script in a Windows machine, but also to improve the transfer of the environment from one device to another or from an Anaconda version to another one.

Analysis of the script. The script we wrote to detect faces involved the OpenCV library and the Haar Feature-based Cascade Classifiers.

OpenCV comes with a trainer as well as detector. If you want to train your own classifier for any object you can use OpenCV to create one. Dealing with detection, it already contains many pre-trained classifiers for face, eyes, smiles, etc. Those XML files are stored in the opencvdatahaarcascades folder.

Moreover, you have often to capture live stream with camera. OpenCV provides a very simple interface to this. Basically, you have to capture a video from the camera, convert it into gray-scale video and display it. To realize it, you had to follow a few steps and run some methods.

First, using `cv2.CascadeClassifier`, we had loaded the classifier we needed: among all, we chose `haarcascade_frontalface_default.xml` to detect faces, `haarcascade_eye.xml` to detect eyes and other two to detect ears, that are `haarcascade_mcs_rightear.xml` and `haarcascade_mcs_leftear.xml`.

Then, you need to create a **VideoCapture** object to capture a video. Its argument can be either the device index or the name of a video file: in our case, it has been the URL network of the live. Device index is just the number that specify which camera is connected; so you can simply pass 0 (or -1) for each one. After that, you can capture frame-by-frame and, at the end, release the capture.

Similarly to the capturing from Camera, you can play a video from a file; you just need to change camera index with video file name. Also while displaying the frame, use appropriate time for `cv2.waitKey()`. If it is too less, video will be very fast and if it is too high, video will be slow.

Thereafter, what you need is to save the video. In order to do it, you had to create a **VideoWriter** object. You should specify the output file name and the

FourCC code. Then, number of frames per second (fps) and frame size should be passed; the last one parameter is `isColor` flag. If it is True, the encoder will expect color frame; otherwise, it works with gray-scale frame.

FourCC is a 4-byte code used to specify the video codec. The list of available codes can be found in fourcc.org; it is platform dependent. Some of them are:

- **In Fedora**: DIVX, XVID, MJPG, X264, WMV1, WMV2. (XVID is preferable. MJPG results in high size video. X264 gives very small size video);
- **In Windows**: DIVX;

Specifically, we used the codec that could allow us to save the live video, both the version with the detection of faces and the one without them.

Next step was detecting a face from an image using the CascadeClassifier we loaded; again OpenCV's CascadedClassifier has made it simple for us as it comes with the function `detectMultiScale`, which serves exactly for this purpose. If it finds a face, it returns a list of positions of them in the form `Rect(x,y,w,h)`; if not, then returns `none`. Some details of its arguments are:

- **image**: the first input is the *grayscale* image;
- **scaleFactor**: this function compensates a false perception in size that occurs when one face appears to be bigger than the other simply because it is closer to the camera;
- **minNeighbors**: this is a detection algorithm that uses a moving window to detect objects. It defines how many objects can be found near the current one before it can declare the face found.

Obviously, there are other parameters which could be used to give full details to these functions.

Just changing the Classifier, we used this function to detect the other part of the face we were interested in.

So, looping over (in a for-loop) the list of faces it returned at the first time, we drew rectangles around faces and circles around eyes on our original colored image, using the following methods: `cv.rectangle()` and `cv.circle()`, setting parameters like image, top-left corner and bottom-right corner of rectangle, color and thickness of the line. Outside it, instead, we drew rectangles around ears.

Larix Broadcaster. When the drone was not available to test his functions and his responses to our work, we needed to find a software for mobile devices that used camera and microphone and allowed us to encode and broadcast what we recorded.

The app we found that was right for our goal is Larix Broadcaster: it is a free application for iOS, Android and Windows Phone capable of streaming live video and/or audio content from mobile device to a media server. It provides all the services we were searching for and seemed interesting to us because, as we read in the information box of the app, it can connect to any media server like YouTube, Facebook Live, Wowza Streaming Engine or any other

capable of RTSP and RTMP input. Moreover, it supports multiple simultaneous connections and guarantees low latency of produced stream.

So, we tested several video resolutions, looking for the one that could reduce at the most delays and interruptions. We select 720x480 as the best, even if produces at least 7 seconds of delay, related to the server nginx and to the algorithm used for the face detection.

There are many steps to follow to realize the transmission: first of all, you need to set the IP address as `rtmp://172.19.52.153/live/prova` in the Connection settings of the app, where `172.19.52.153` is the IP of the device that acts as a server.

Next, you had to start nginx and begin to record using Larix Broadcaster. If you want to check if the server is running correctly, just open the tab Processes in Task Manager and look for it.

Instead, if you want to check if the live streaming is working properly, you can use VLC media player: open the Media menu, then the option Open Network Stream and enter the network URL previously generated in the window that will appear. Then, just click on the Play button: if you can watch the video streaming, it means that all the steps have been completed in the right way.

4 Conclusions

In this thesis, we discussed the problem of face detection during a live video streaming recorded by Phantom 4 Pro+.

Fig. 6. Image selected from the live video showing different types of detection

It interested us because of the great demand for this kind of identification in security services, agencies, investigative services and so on.

In fact, the human face plays an important role in our social interaction, conveying people's identity. Using the human face as a key to security, biometric face recognition technology has received significant attention in the past several years due to its potential for a wide variety of applications in both law enforcement and non-law enforcement.

As compared with other biometrics systems using fingerprint or palmprint and iris, face recognition has distinct advantages because of its non-contact process. Face images can be captured from a distance without touching the person being identified, and the identification does not require interacting with the person. In addition, face recognition serves the crime deterrent purpose because face images that have been recorded and archived can later help identify a person.

This, in addition to the progress of technologies, led to the use of new devices, such as drones, even for face detection.

So, the aim of the present project became to realize a new architecture that could help us to distinguish faces and its features, especially eyes and ears, while filming a live streaming by the above-mentioned aircraft (you can see first picture of this paragraph - Fig.6 - and the following images - Fig.7, Fig.8 - to have an idea of how it really works).

Fig. 7. Image selected from the live video showing different types of detection

As you can see in these images, while the drone is filming some people (members of our group), the script is running and getting results (displayed on the computer).

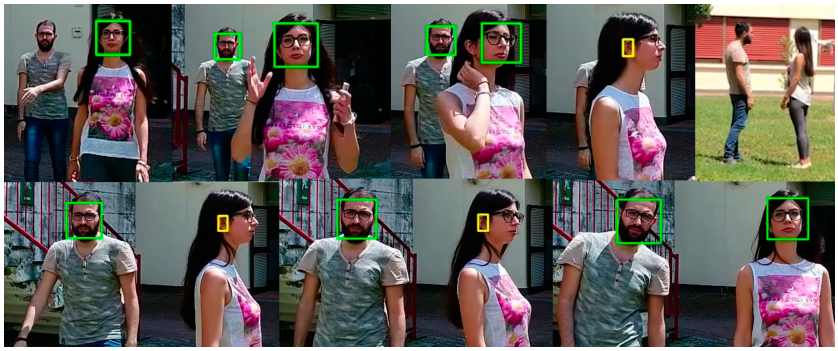

Fig. 8. Image selected from the live video showing different types of detection

Besides, we had also estimate some significant values: the greatest distance required to detect face is about 8 meters, while the greatest one required to detect eyes and ears is about 1 meter (see Fig.9). Moreover, the resolution chosen for the streaming of the drone is 1280x720, which is also the smallest it supports, while the one chosen for smartphones (using Larix Broadcaster) is 720x480.

Fig. 9. Comparison between drone and computer visualization

Honestly, there is a little delay in the execution of the detection algorithm and in its visualization because of the high number of components that are working together (see Fig.10). In fact, nearly 5-7 seconds of delay are due to the chosen server and they can vary according to the quality of the connection;

then, at least 2 seconds are required by the algorithm, but they can increase depending on the processing of the images and on calculating capacity. So, we have a total delay of about 7-8 seconds; however, we can still consider it as a real-time system, considering that we had worked using our average computers, but, if you have the opportunity to test it with a more powerful pc, you will sure notice the difference.

Fig. 10. Comparison between drone and computer visualization. More in details, the first picture of the second row is extracted from the video streaming on computer, while the next one represents the same instant processed by the script

The created system needs some important steps to be executed correctly: first of all, a server (in our cases nginx) has to be started. Then, you obviously need a live video recorded by Phantom 4 Pro+, whose capabilities of the camera are one of its best and most underrated features. The sensor is good, but the video capture speeds it can reach are better. 4K video comes in the form of standard 3840x2160 resolution or Cinematic 4K of 4096x2160, both at up to 60 fps, but we had to reduce it to 720x480 during tests to avoid increasing the delay between the tracking and the visualization on the display. In fact, the next stage consists in associating the drone with the nginx server: in order to do it, make sure both Phantom 4 Pro+ and the computer are connected to the same network; so, find out the IP address of the device working as the server and put it as the URL network required to reproduce live streaming using RTMP protocol

in the "Choose platform" configuration of the drone. At this point, both the devices are connected and the stream of the detection can begin: if it works, you would see rectangles and circles of different colours surrounding faces, eyes and ears of the framed people.

4.1 Future work

In view of future works and researches in the same area of interest, there are several updates that could be done.

First, it could be helpful to add a grid on irises in order to identify them better and faster; keeping on thinking about eyes, it could be interesting to improve the precision in their recognition even if a person is far from the camera. For this reason, circles around eyes should modify their size adapting to the face on which they are drawn; in fact, now it is a fixed parameter.

Furthermore, it would be useful to implement methods to provide face recognition in addition to the detection. For this reason, feature extraction should be improved and mainly tested, because it is crucial for any recognition algorithm and system. A remarkable change would been noticed in the recognition rate using different feature extraction. Specifically, in cropping an image before it is run through a recognition system, there would be still much work to be done in this area. It would be interesting to explore new techniques of preprocessing that would lead to the optimal recognition rates. In this case, face and ear biometrics has been used, but perhaps there are other metrics that can be combined that will lead to a more robust system. It would be interesting to see what kind of results could be achieved in a system like this with different metrics.

References

1. Viola, P., Jones Michael J.: Robust Real-Time Face Detection. In: International Journal of Computer Vision **57**(2), (2004).
2. Viola, P., Jones Michael J.: Rapid Object Detection using a Boosted Cascade of Simple Features. In: Accepted Conference on Computer Vision and Pattern Recognition, (2001).
3. Freund Y., Schapire R. E.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Journal of computer and system sciences **55**, pp. 119–139, (1997).
4. Papageorgiou C., Oren M., Poggio T.: A general framework for object detection. In: International Conference on Computer Vision, (1998).
5. Freund Y., Schapire R. E., Bartlett P., Lee W. S.: Boosting the margin: a new explanation for the effectiveness of voting methods. Ann. Stat., **26**(5), pp. 1651–1686, (1998).
6. RTMP Wikipedia page, https://en.wikipedia.org/wiki/Real-Time_Messaging_Protocol. Last accessed August 2014.