**An Anti-Spoof Face detector**

P.Sandun Dimuthu Dilshan Fonseka

(HF17A1SEBENG)

BEng (Hons) Software Engineering

Staffordshire University, UK

A project report submitted in partial fulfilment of the

BEng (Hons) Software engineering award

Supervised by Mr. Gamindu

Assessed by

July, 2019

Abstract

This report is about an anti-spoof detector implemented using python language. The report contains and vital information about current spoofing techniques. Biometrics, background of haar cascades, computer vision, viola jones algorithm. Eye blink detection. Information about eye aspect ratio. The goal of this report is to detect the liveness of a subject in front of a video feed and identify if the user is a spoof or not.

Acknowledgement

*I thank God for giving me this opportunity and helping me see it through.*

I would like to thank my supervisor who helped me turn this project idea into a real world solution

I would like to extend thanks to the project Manager who helped to prioritize and manage my idea in a formal manner.

I would like to extend my gratitude to APIIT for allowing me to complete this project. By providing me with the necessary environment and technical equipment.

Furthermore I would like to mention my colleagues and my parents for helping me in completing this assignment

Finally I take the responsibility for any errors in the documentation

Table of contents

[Chapter 1. Context and Preliminary investigation 6](#_Toc15029658)

[Chapter 1.1 Background 6](#_Toc15029659)

[Chapter 1.2 Biometric origins 7](#_Toc15029660)

[Chapter 1.3 Introducing Biometrics 8](#_Toc15029661)

[Chapter 1.4 Choice of investigation 10](#_Toc15029662)

[Chapter 1.4.1 Spoofing techniques 10](#_Toc15029663)

[Chapter 1.4.2 Identifying liveness to prevent spoof attacks 11](#_Toc15029664)

[Chapter 1.5 Scope and objectives 13](#_Toc15029665)

[Chapter 1.5.1 Goals and objectives 13](#_Toc15029666)

[Chapter 1.5.2 Assumptions 14](#_Toc15029667)

[Chapter 1.5.3 Constraints 14](#_Toc15029668)

[Chapter 1.5.4 Deliverables 14](#_Toc15029669)

[Chapter 1.6 Literature review 15](#_Toc15029670)

[Chapter 1.6.1 Background 15](#_Toc15029671)

[Chapter 1.6.2 Haar Features 16](#_Toc15029672)

[Chapter 1.6.3 Viola jones algorithm 17](#_Toc15029673)

[Chapter 1.6.4 Eye blink detection 18](#_Toc15029674)

[Chapter 1.6.5 Eye Aspect Ratio? 19](#_Toc15029675)

[Chapter 1.6.6 Conclusion 20](#_Toc15029676)

[Chapter 1.7 Project plan 21](#_Toc15029677)

[Chapter 2. Analysis 22](#_Toc15029678)

[Chapter 2.1 Choosing the Anti Spoofing techniques 22](#_Toc15029679)

[Chapter 2.2 Table of comparison 23](#_Toc15029680)

[Chapter 2.3 Analysis conclusion. 25](#_Toc15029681)

[Chapter 2.4 System requirement specification 26](#_Toc15029682)

[Chapter 2.4.1 Functional requirements 26](#_Toc15029683)

[Chapter 2.4.2 Non-functional requirements 26](#_Toc15029684)

[Chapter 3. Design 27](#_Toc15029685)

[Chapter 3.1 Overall Activity diagram 27](#_Toc15029686)

[Chapter 3.2 Detect face 29](#_Toc15029687)

[Chapter 3.3 Detect eyes 30](#_Toc15029688)

[Chapter 3.4 Detect smile 31](#_Toc15029689)

[Chapter 3.5 Detect facial landmark 32](#_Toc15029690)

[Chapter 3.6 Detect eye blink 33](#_Toc15029691)

[Chapter 3.7 Class diagram 34](#_Toc15029692)

[Chapter 3.8 User interface 36](#_Toc15029693)

[Chapter 4. Implementation 37](#_Toc15029694)

[Chapter 4.1 OpenCV, Python and Dlib 37](#_Toc15029695)

[Chapter 4.2 Detecting face, eyes and smile 38](#_Toc15029696)

[Chapter 4.3 Detecting facial landmarks 41](#_Toc15029697)

[Chapter 4.4 Detect eye blink 42](#_Toc15029698)

[Chapter 5. Testing 45](#_Toc15029699)

[Chapter 6. Critical Appraisal 49](#_Toc15029700)

[Chapter 6.1 what did I learn? 50](#_Toc15029701)

[Chapter 6.2 Limitations 50](#_Toc15029702)

[Chapter 6.3 Overcoming limitations 51](#_Toc15029703)

[Chapter 6.4 Commercial product? 51](#_Toc15029704)

[Chapter 6.5 Weaknesses and how to overcome them! 51](#_Toc15029705)

[References 52](#_Toc15029706)

Table of figures

[Figure 1 (Computer vision, 2018) 17](#_Toc15030384)

[Figure 2 (Computer vision, 2018) 17](#_Toc15030385)

[Figure 3 (Computer vision, 2018) 18](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030386)

[Figure 4 (Computer vision, 2018) 18](#_Toc15030387)

[Figure 5 (Rosebrock, 2019) 20](#_Toc15030388)

[Figure 6 (Rosebrock, 2019) 20](#_Toc15030389)

[Figure 7 Overall flow of activity diagram 29](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030390)

[Figure 8 Detect face activity diagram 30](#_Toc15030391)

[Figure 9 Detect eyes activity diagram 31](#_Toc15030392)

[Figure 10 Detect smile activity diagram 32](#_Toc15030393)

[Figure 11Detect Face landmarks 33](#_Toc15030394)

[Figure 12Eye blink activity diagram 34](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030395)

[Figure 13Class diagram 35](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030396)

[Figure 14 Code snippet 39](#_Toc15030397)

[Figure 15Code snippet 39](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030398)

[Figure 16Code snippet 40](#_Toc15030399)

[Figure 17(Medium, 2019) 40](#_Toc15030400)

[Figure 18Code snippet 41](#_Toc15030401)

[Figure 19code snippet 41](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030402)

[Figure 20Code snippet 41](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030403)

[Figure 21Code snippet 42](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030404)

[Figure 22Code snippet 42](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030405)

[Figure 23Code snippet 42](#_Toc15030406)

[Figure 24Code snippet 42](#_Toc15030407)

[Figure 25Test 1 snapshot 46](file:///C:\Users\Beast_\Desktop\Anti%20Spoof%20Detector%20FYP.docx#_Toc15030408)

[Figure 26 Test 2 snapshot 46](#_Toc15030409)

[Figure 27Test 3 snapshot 47](#_Toc15030410)

[Figure 28Test 4 snapshot 47](#_Toc15030411)

[Figure 29Test 5 snapshot 48](#_Toc15030412)

[Figure 30Test 6 snapshot 48](#_Toc15030413)

[Figure 31Test 7 snapshot 49](#_Toc15030414)

[Figure 32Test 8 snapshot 49](#_Toc15030415)

# Chapter 1. Context and Preliminary investigation

## Chapter 1.1 Background

Humans are unique beings. Our fingerprints are never the same, certain people have certain behavioural aspects which make them special. The iris, faces, fingerprints and the sound of voices are some aspects of people which make them quite different from one another, with the aid of biometrics we are now able to uniquely recognize people from these behavioural characteristics according to (Ross et al., 2006). The level of authentication’s sophistication has reached to new heights due to biometric authentication making a huge impact in our lives in the modern world. We even now use these biometric authentication techniques on our smartphones. Biometrics in short is ***something we are*.** According to (Ross, Nandakumar and Jane, 2006) even an authentication methodology which we assume is so secure can even be tricked or spoofed.

There are many biometrics which needs to be reinforced. This paper will mainly focus on the face biometrics of a human. In regard to this biometric aspect. Face authentication is a popular form of authentication in our day to day lives. Face authentication can be spoofed using three mainly fatal ways. These include

1. By using photographic material
2. A video material
3. Finally a 3D model of the user

The goal here is to identify when a user is trying to fool a face authentication using these 3 fatal techniques to spoof attack a face authentication system.

## Chapter 1.2 Biometric origins

Scientific biometrics emerged to the surface in the 1880’s according to (Office for science Government, 2018). The goal was to provide safer authentication for employment contracts. Furthermore the biometric systems were required in order to identify repeat offenders. Who are people who did crime and they could be identified using their fingerprints. In the beginning when storing biometric data it was mainly body measurements and fingerprints. The latter which are fingerprints was the authentication methodology that became famous due to its ease of enrolling and authenticating apart from the body measurements according to (Office for science Government, 2018). This has provided to be true, regarding the 21st century humans live in. People now have biometric locks on their mobile devices. The biometric technology has evolved into a much easier form of authentication than even passwords.

(Office for science Government, 2018) mentions that biometrics have become a go-to access control mechanism. Nowadays biometric solutions provide protected secretive access to a person’s house, vehicle, workplace, financial data and travel rights. Furthermore all these solutions involve not just biometrics, which is just not something you are but also with something you have and something you know. This involve the multi factor authentication system, which is even better. Now what separates the biometric authentication from the rest of the authentication methodologies, is that data which is something you know and something you have can be shared, or even stolen, but something you are. Which is biometrical data cannot be stolen or shared. This is why biometric authentication is very important and secure.

## Chapter 1.3 Introducing Biometrics

What is biometrics? If we do breakdown the word it is *bio-metric.* The first half of the word bio meaning biology and the latter part of the word being metric which is also known as form of average measurement. According to (Office for science Government, 2018), biometrics is the point where statistical methods meet biological data. Which is factually correct, but why would we require such advanced form of authentication, and what is authentication.

According to (Bolle, 2011), authentication is when an individual proves an identity. In a simpler term a successful authentication proves that you are who you say you are. (Bolle, 2011) mentions that there are three traditional methods of authentication. Which are namely. ***Possessions***, ***knowledge, biometrics***.

1. Possessions involve the idea of **something you have** it could be keys, passport id, identity card even smart cards. What is a smart card? A smart card is tangible, electronic based authorization scheme. This smart cards contain a built in microprocessor. Smart cards are used in banking, id verification access control at workplaces and even used in health care applications where a user’s vital details are stored in a smart card for safety and reliability.
2. Knowledge involve the idea of **something you know**, these information need to be kept in secret. This involves authentication methods like passwords, pin security, pattern security. Only the owner of this information must know these vital material.
3. Biometrics involves the use of **physiological and behavioral appearances** or **even individual characteristics** that makes each person unique. Characteristics which involve human body parts and human actions, according to (Bolle, 2011). The six more regularly used biometrics are mentioned below

|  |  |
| --- | --- |
| *Physiological* | *Behavioral* |
| Face | Signature |
| Fingerprint | Voice |
| Hand geometry |  |
| Iris |  |

Apart from the six normally used biometric data. There are also 9 other less frequently used or even in the stages of research biometrical data displayed below in tabular fashion.

|  |  |
| --- | --- |
| *Physiological* | *Behavioral* |
| DNA | Gait |
| Ear shape | Keystroke |
| Odor | Lip motion |
| Retina |  |
| Skin reflectance |  |
| Thermogram |  |

## Chapter 1.4 Choice of investigation

Biometrics has continued to emerge into light. Nowadays we have multiple authentication methodologies. The most frequent methodologies in the light are face, fingerprint, iris and voice according to (Ross et al., 2006). These multiple techniques are great for authentication because it is something the user is, but even these authentication methods can be tricked according to (Ross, Nandakumar and Jane, 2006). This report will focus mainly on face biometric, and how spoofing is achieved in face recognition, and how to identify spoof attacks on face authentication systems.

### Chapter 1.4.1 Spoofing techniques

There are three main ways a spoof attack can be introduced to a face authenticator system.

1. Photographic material of a registered user

A photo of a registered user is downloaded or captured and then presented to the face authentication system. Getting a facial image of a certain user is fairly an accessible method. The photo which is physical material can be bent and moved to fool the system into thinking the photo is a live person.

1. Video material of a registered user

A video of the user is presented to the camera. Where the facial movements can be fooled to provide evidence that the user is a live person

1. 3D model of a registered user

The 3D model is created according to the registered person’s facial features. The authenticator can be fooled. This option though effective is quite expensive.

### Chapter 1.4.2 Identifying liveness to prevent spoof attacks

1. Facial expression

The face of the user is captured. The image is preprocessed and identified the user’s facial expression. These facial expression are identified using the eyebrows, eyes, mouth.

1. Depth information

Stereovision is one way to get the depth information from a human subject using a lens.

To get the depth information it uses stereo image pairs with the help of two cameras to produce a certain disparity map. Disparity map define the number of pixels or the motion between the pair of stereo images obtained. The disparity maps are converted into depth maps. This method is quite expensive. The depth value of a real face has a much bigger varied range than that of a plant face image.

1. Mouth movement

Requires the user to move his/her mouth in order to continue with the face detection. The people were recorded speaking digits from 0-9 the aim of this scenario is to recognize the digits by the lip motion only to indicate liveness

1. Head movement

Requires a motion video. The head movement of the subject is captured. If the head movement is carried out then the user can be regarded as a live user.

1. Eye blinking

Firstly both eyes are detected. The face regions are normalized and the eye regions are extracted. Since eye blinking is a physiological activity. A person approximately blinks their eyes once every 2 – 4 seconds so the user collaboration is almost nonexistent.

1. Multi-model

The multi model is a collaboration of both the face and voice. The face is detected along with a voice recorder. The user should speak and the according to the mouth movements the voice should be recognized. This requires extra hardware and more user collaboration

1. Facial Thermogram

Facial Thermogram detect the heat of the subject in front of the camera. For this special cameras needs to be used. High end cameras are expensive. Less user collaboration is required

1. Interactive response

The system asks the user to perform a certain activity according to the actions performed by the user. The system detects if the user is a live subject or not. This is very effective but the user collaboration is very high.

## Chapter 1.5 Scope and objectives

### Chapter 1.5.1 Goals and objectives

The project scope for this assignment and deliverables are displayed below in detail

* Anti-Spoof Face detector

1. Detect human faces
2. Detect face changes
3. Detect eye changes
4. Detect the spoof attack

Goal 1: Detect human faces

* Objectives

1. The user’s face should be detected when he is in front of the camera
2. The user’s eyes along with the face should be detected

* Deliverable
* A successful face detection
* A successful eyes detection

Goal 2: Detect face changes

* Objectives

1. The eye blink should be detected
2. The number of times the eye blinked should be displayed
3. The face should be detected if movement is seen

* Deliverable
* A successful eye blink detection
* An increasing counter for eye detection
* Ability for the program to identify a face movement

Goal 3: Detect eye changes

* Objectives

1. The eye blinks should be detected to check if the person is a live person

* Deliverable
* A successful eye blink detection

Goal 4: Detect spoof attack

* Objectives

1. Ability to detect if the eye is not blinking
2. Ability to detect if the face movement is not there
3. Ability to detect if the presented evidence is photograph or a video

* Deliverable
* Detect the spoof attack

### Chapter 1.5.2 Assumptions

* Assuming the user has a computer and is able to run the program
* The user should have proper lighting in order to use the program to its full effect
* The user should have a decent web cam which can capture proper feed
* The user should place themselves correctly in front of the camera

### Chapter 1.5.3 Constraints

* A 3D mask test would be quite difficult to test since it is expensive to carry it out.

### Chapter 1.5.4 Deliverables

* Anti-Spoof Face detection software

1. Detect human faces
2. Detect face changes
3. Detect eye changes movement
4. Detect the spoof attack

## Chapter 1.6 Literature review

### Chapter 1.6.1 Background

For an idea to come to life a research is quite a vital aspect of the project. The literature which is research plan for this area is mentioned below.

There are four goals to achieve

1. Detect face
2. Detect eyes
3. Detect eye blink
4. Detect spoof

In order to detect a human face. The most mainly used library is, the open source computer vison library also known as OpenCV. OpenCV has multiple wrapper classes which can be integrated with multiple languages. In order to detect objects in a face, haar cascades are required. To understand how haar cascades work the user needs to understand how the viola jones algorithm works. In order to complete these four goals successfully the user will have to research into. A method to detect user’s face, user’s eyes. Once the face and eyes can be detected the eye blink can be detected. Whereby the system will be able to identify if the subject is a spoof or not. The user will gain enough knowledge and will lead the user to understand the situation of the each goal and assess the situation with ease since these topics have been researched.

### Chapter 1.6.2 Haar Features

Haar wavelet was introduced by Alfred Haar in 1909. Haar wavelets uses edge features and line features. Haar wavelets is similar to a sequence of square shaped functions.

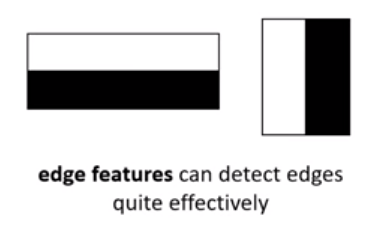
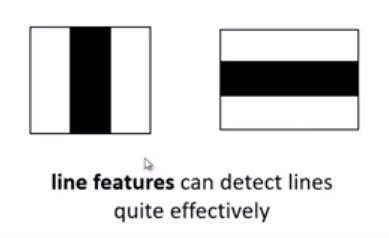
 

Figure 1 (Computer vision, 2018)

In order to detect faces. There are certain similarities in human faces such as eyes, nose, and mouth. By using these square shaped sequences above it is possible to detect these similar components in everyone’s faces. It is all about the pixels. To accurately identify a face the colour images need to be converted to grey images in order to identify black pixels from the white pixels according to (Junaidy, Wulandari and Tanudjaja, 2019).

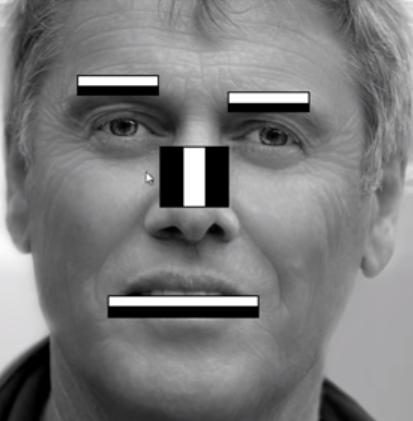
 

Figure 2 (Computer vision, 2018)

In the above picture its displayed how the most relevent features in the human face are recognized by computer vision.

Eyebrows – The top part of the eye brow as displayed is white pixels in regard to the bottom part which is black pixels

The nose area – The middle part of the nose as displayed is white pixels and the surrounding parts are black pixels

The mouth – The teeth area is white pixels where the lips area is black pixels

### Chapter 1.6.3 Viola jones algorithm

A grey image of the human face, is not entirely black and white. Both black and white may have a palette of colours ranging from very dark to mild dark, or even very white to mildly white.

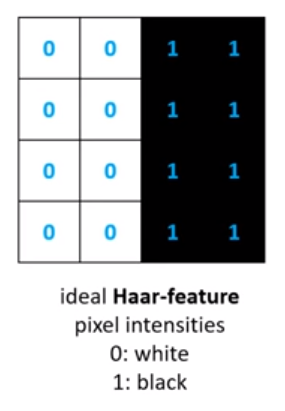


Figure 3 (Computer vision, 2018)

So displayed above is are how the dark and white pixel intensities are given a value. In order to understand how the haar like features are confirmed we need to understand the Viola Jones algorithm.

The Viola Jones algorithm will need

1. The average of the white pixels
2. The average of black pixels

Of a gray image.

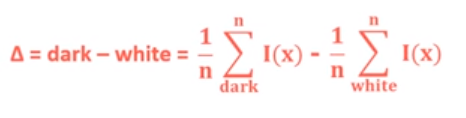


Figure 4 (Computer vision, 2018)

An ideal haar feature is given the value of 1. This happens in 3 steps

1. Get the average of the dark pixels
2. Get the average of the white pixels
3. Substract the white pixels from the dark pixels in order to get a value closer to 1

The closer the value to 1. That is when a haar like feature is recognized according to (Junaidy, Wulandari and Tanudjaja, 2019).

### Chapter 1.6.4 Eye blink detection

An eye blink is a rapid opening and closing of the eye, this action could be voluntary or even involuntary. Most of the time it is an involuntary human action. An eye blink lasts for 100-400 ms. Eye blink detection is a very important method used in applications such as.

* Driver drowsiness detection
* Anti-Spoof face recognition systems
* Communication systems for disabled

There are active and passive ways to calculate the eye blinks of a certain user in the software application. The active ways are quite expensive since it requires special hardware. The passive ways are in comparison cheaper and would only require a standard camera in order to gain a proper video feed. Both face and eyes are identified using a viola-jones type detector according to (Cech, 2016).

In order to calculate the eye blink detection there should be a way to localize the user’s face and track the user’s facial landmarks. Including the eyes and the eye lids according to (Cech, 2016).

How to calculate an eye blink?

1. Localize the face especially eyes and eye lids
2. Calculate the Eye aspect ratio
3. If the value from the eye aspect ratio reaches a certain criteria the eye blink can be detected

### Chapter 1.6.5 Eye Aspect Ratio?

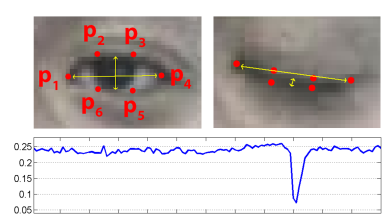


Figure 5 (Rosebrock, 2019)

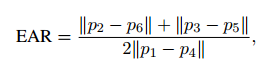


Figure 6 (Rosebrock, 2019)

In order to calculate the eye aspect ratio we need height and width of the user’s eye. Whenever the camera is working and the video feed provides a user in front of it. The user’s facial landmarks are detected and tracked. These points are then used to calculate the eye aspect ratio of the user according to (Cech, 2016).

In order to calculate the eye aspect ratio. There are two points in a user’s eye which needs to be calculated in order the get the height of the user’s eye. This is done by

1. Getting the length between p2 and p6
2. Getting the length between p3 and p5
3. Then we add these two values

Next according to the equation. We calculate the width of the user’s eye. This is done by

1. Getting the length between the points p1 and p4 and multiply this value by 2

Finally the addition of the 2 points which was used to calculate the height is divided by the width which is multiplied by 2. This gives us the eye aspect ratio.

As you can see in the graph, the eye aspect ratio value is above zero (0) when the user’s eye is open, but it plummets to zero (0) when the user’s eye is closed. This is how to identify an eye blink according to (Cech, 2016).

### Chapter 1.6.6 Conclusion

The four goals mentioned in the background of this report can now be achieved with the information found in this report.

|  |  |  |
| --- | --- | --- |
| Goal | Achievement | Information from where to find and how it works? |
| Detect face and eyes | Can detect face and eyes using haar cascades. | Ability to understand haar features and viola jones algorithm in how the detection works was achieved and mentioned in this report |
| Detect eye blink | Using the eye aspect ratio | Instead of using image processing the eye aspect ratio is a very efficient way to calculate an eye blink of the user. |
| Detect spoof |  | With the information found and researched during this project. A spoof can be detected by the number of eye blinks of the user. |

## Chapter 1.7 Project plan

The methodology followed for this assignment is the agile methodology. The reason for choosing this methodology is because in this project in the Chapter 1.5.1 all the goals are divided in four main aspects. Each aspect will be assigned a sprint. Each sprint will lead to the continuous development and implementation of the project.

|  |  |  |  |
| --- | --- | --- | --- |
| Goal | Description | Difficulty level | Duration |
| Detect human faces | The system should be able to detect human faces | Easy | June 2nd 2019 – June 7th 2019  **6 DAYS** |
| Detect eyes | The system now should be able to detect eyes | Medium | June 7th 2019 – June 14th 2019  **7 DAYS** |
| Detect eye blink | The system should be able to detect eye blink | Hard | June 14th 2019 - June 30th 2019  **16 DAYS** |
| Detect spoof | The system should be able to detect if the user is a spoof or not | Hard | June 14th 2019 - June 30th 2019  **16 DAYS** |

# Chapter 2. Analysis

## Chapter 2.1 Choosing the Anti Spoofing techniques

Liveness detection is the keyword related closely with anti-spoof. Depending on the liveness of the individual the system will have to identify if the user is a spoof or not.

There are three main techniques

1. Texture detection

Checks the user’s current given input against information stored preferably a snapshot taken during system sign up.

* This idea is less expensive to implement
* The texture of the photo and the original user should be detected and identified.

1. Motion detection

This method quite difficult to spoof since the motion is detected of the subject. A photographic material provided as input will clearly be rejected. The user motion will be detected, the system will look for a head movement, eye blink.

* Requires a real time video preferably a decent web cam
* The brightness should be controlled

1. Life signs

This is method is very effective. Than both the options mentioned above and it’s very difficult to spoof. The subject will be asked to open or close their left eye. Move head or tilt. Blink eyes in order for the system to identify if the subject is a spoof or not.

* Quite expensive to implement
* User collaboration is vital

**Conclusion of choice** *– Clearly all three choices are great. The system however would be vulnerable if it was just a texture detection or motion detection. The life sign detection is also quite expensive to implement. The choice is to go for the number 2, which is motion detection and also number 3, because if we can choose motion detection and still control it with certain parameters, for example eye blinks for 20 seconds. We can identify spoof, the final choice is 2 and 3.*

## Chapter 2.2 Table of comparison

The methods mentioned here are also mentioned in detail in chapter 1.4.1. 3 factors are considered in this table

1. If data quality is high this means it requires expensive hardware.

* Low – less expensive hardware required to implement
* Moderate – moderate level expensive hardware required to implement
* High – Very expensive hardware required to implement

1. When it means no additional hardware is required this means that only a web camera is required to do the job.
2. User collaboration is also in 3 levels

* Minimal – Almost nothing user is supposed to do just stand or stay in front of camera
* Moderate – User will have to blink or do some action and liveness will be detected from these actions
* Maximum – The system will ask user to do specific actions at specific times which could be quite not user friendly

|  |  |  |  |
| --- | --- | --- | --- |
| *Texture based* | | | |
| Facial expression | Data quality should be high | No additional hardware required | User collaboration is moderate |
| Depth information | Data quality should be high | No additional hardware required | User collaboration is minimal |
| Facial Thermogram | Data quality depends on hardware | Requires IR based camera | User collaboration is minimal |
| *Motion based* | | | |
| Facial expression | Data quality should be high | No additional hardware required | User collaboration is maximum |
| Mouth movement | Even moderate data quality can be dealt with | No additional hardware required | User collaboration is moderate |
| Head movement | Data quality should be high | No additional hardware required | User collaboration is maximum |
| Eye blinking | **Even low data quality can be dealt with** | **No additional hardware required** | **User collaboration is minimal and almost non existent** |
| *Life sign based* | | | |
| Multi model | Data quality depends on hardware | Additional hardware is required | User collaboration is maximum |
| Interactive response | Data quality depends on hardware | No additional hardware required | User collaboration is maximum |

## Chapter 2.3 Analysis conclusion.

During this analysis stage we understood three main types of anti-spoofing techniques. Which are

1. Texture detection
2. Motion detection
3. Life sign detection

All three aspects were considered and tabulated in chapter 2.3. The conclusion to the choice came easily.

* Eye blink detection

The eye blink detection is the only form of detection where not high end expensive equipment was not required. It is the only process where user collaboration was the least, and where no additional hardware was required. Due to these reasons eye blink detection is the clear choice.

## Chapter 2.4 System requirement specification

The system requirement specification gives a clear picture to the developer regarding which features to implement in the system. The requirement specification is discussed with the client and agreed upon before implementation starts.

### Chapter 2.4.1 Functional requirements

1. Detect face – When the user is in front of the camera. The software should detect the user using the web camera preferably
2. Detect eyes – After the user’s face is successfully detected then the user’s eyes should be detected
3. Detect eye blink – The user’s eye blink should be detected
4. Detect face movement – The user’s face movement should be detected after both the user’s face and eyes have been detected
5. Detect spoof – If certain criteria such as the detection of the face and eyes. Including the dynamic changes in the face are not detected then the system should identify the user as a spoof.

### Chapter 2.4.2 Non-functional requirements

1. User friendly – The system should be easy to understand. The system should provide clear commands to the user. The system should follow the UI principles.
2. Performance – The system should be able to quickly identify the spoof attack.
3. Accuracy – This attribute to the system runs parallel with performance. The accuracy of the system should be as good as the speed of the system.
4. Exception handling – The system should be able to handle exceptions and not let the system run into errors.

# Chapter 3. Design

This is the designing stage of the software application, in this stage a clear design is created in order to model the application how the user wants to. After the design stage where every flow of how a function should work is mentioned developers can start working on the implementation by understanding the design diagrams and models.

## Chapter 3.1 Overall Activity diagram

The application starts with user showing his/her face. Application detects the face. Localizes the face, then continues to detect the eyes, in order in regard to the activity diagram shown below the face movement and eye movement is checked for blinks or head swivel. If these conditions are met a spoof is identified at any point of the flow in the activity diagram, if one function is not completed an error message will be prompted to the user.

|  |  |
| --- | --- |
| Type | No of actions |
| Activity diagram | 10 (including the error flow) |
| MAIN FLOW WITHOUT ANY ERRORS   1. Start application 2. User shows face 3. Detect face 4. Detect eyes 5. Detect eye movement 6. Detect face movement 7. Identify spoof | |
| 3a. If face not detected display error message  4a. If eye not detected display error message  5a. If face movement not detect display error message  6a. If eye movement not detected display error message  7a. If spoof identified display error message  7b. If spoof not identified display “passed” message | |

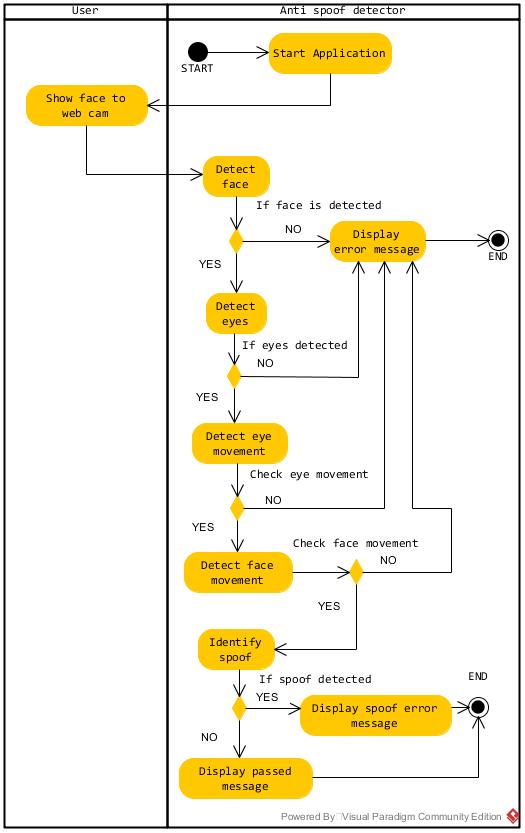


Figure 7 Overall flow of activity diagram

## Chapter 3.2 Detect face

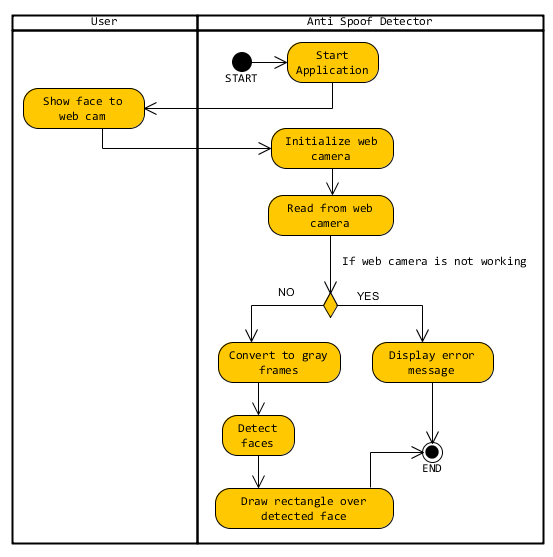


Figure 8 Detect face activity diagram

## Chapter 3.3 Detect eyes

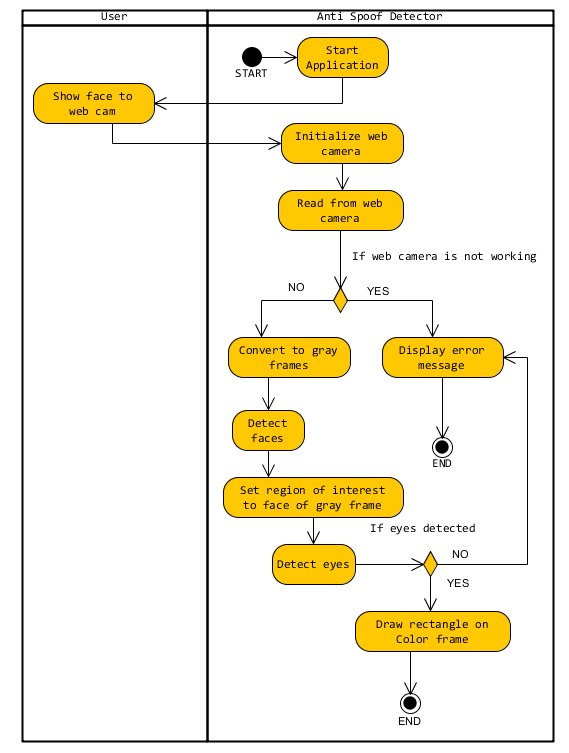


Figure 9 Detect eyes activity diagram

## Chapter 3.4 Detect smile

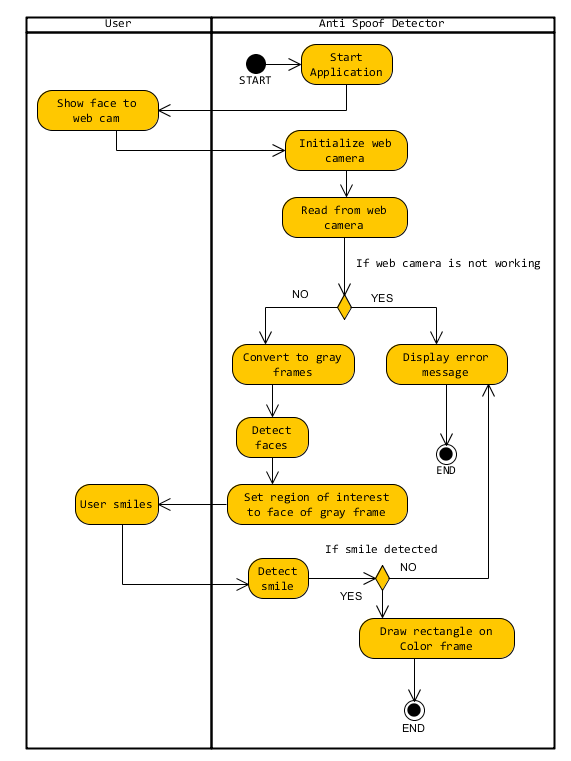


Figure 10 Detect smile activity diagram

## Chapter 3.5 Detect facial landmark

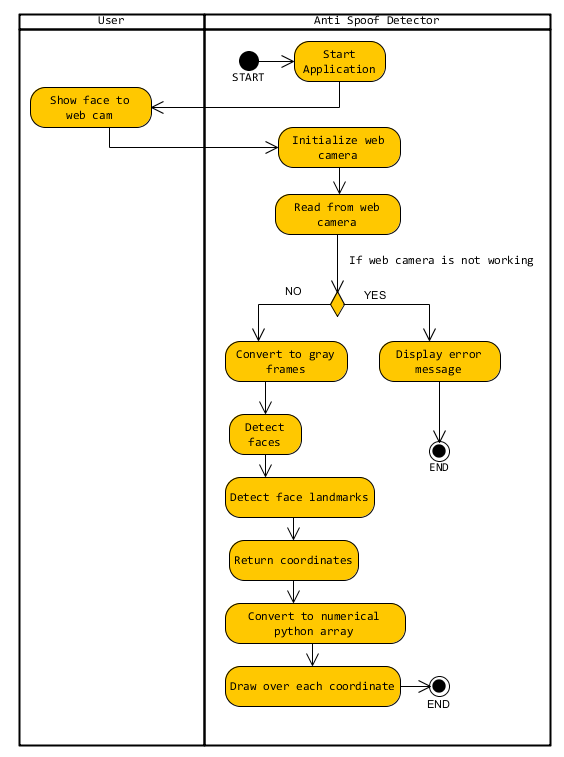


Figure 11Detect Face landmarks

## Chapter 3.6 Detect eye blink

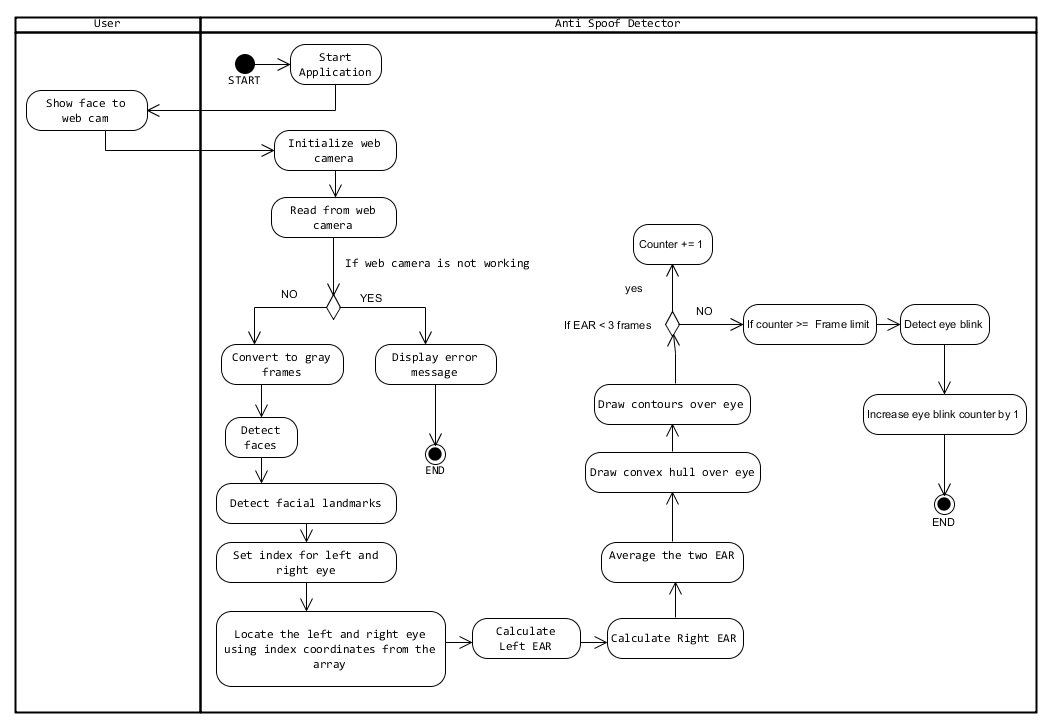


Figure 12Eye blink activity diagram

## Chapter 3.7 Class diagram

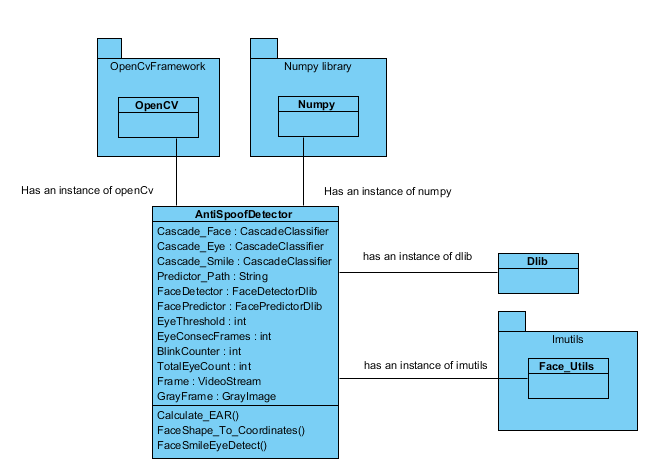


Figure 13Class diagram

The AntiSpoofDetector is the main class here. This class has instances of all the libraries connected around this class. They are OpenCV, NumPy, Dlib and Imutils.

1. OpenCV

* Used for making use of the haar cascades which is involved in detecting face features such as eyes face and smile.
* Used for converting the RGB frames to gray images
* Used for drawing contours, circles and rectangles over facial features and displaying out put on the window frame for the user to see.

1. NumPy

* NumPy is a library used to convert the facial landmark coordinates into a two dimensional array also known as numerical python

1. Dlib

* Dlib is used to get the facial landmark points of the face and to make use of its frontal face detector in the process.

1. Imutils
   * Is used to get the indexes of the left and right eye in order to calculate EAR

Variables

* + The Cascade\_Eye, Cascade\_Face and Cascade\_Smile are used to hold the haar cascades which are xml files in the form type of Cascade Classifier (which is in the OpenCV class)
  + The Predictor path variable holds the path for the landmark detection which is a string. This file is a .dat file
  + The Face Detector and Face Predictor variables hold the instances from Dlib
  + The Eye Threshold, Eye Consec frames, Eye blink counter and the total eye blinks are used to calculate the eye blinks these in the data type of int
  + Frame holds the video stream from the web camera, this file type is video stream and is in Rgb
  + GrayFrame variable is used to hold the video frames after the Rgb frame has been converted to gray scale image.

Methods

* + The Calculate EAR method calculates the eye aspect ratio
  + The Face Shape to Coordinates method converts the face coordinates into a two dimensional array with the help of the NumPy library
  + Face Eye Smile detect method detects the eye face and smile when the user is in front of the web camera

## Chapter 3.8 User interface

Displayed below is the user interface. No buttons are involved in this interface.

1. When the face is detected the system will print a face detected message at index number 1
2. When the eyes are detected the system will display that at index number 2
3. When the smile is detected the system will display that at index number 3
4. The eye blink counter will be displayed at index number 4
5. The eye aspect ratio will continuously be displayed and value changed and displayed accordingly at index number 5
6. System will display if the user is a spoof or not at index number 6

Text here 1

Text here 2

Text here 3

Text here 4

Text here 6

Text here 5

The user will be displayed on this window

# Chapter 4. Implementation

## Chapter 4.1 OpenCV, Python and Dlib

It is an open source package or even a library, which can be used in the area of computer vision. The library can aid languages such as python, java, C++. OpenCV is free for all according to (Bradski and Kaehler, 2012).

Main applications for OpenCV includes

1. Facial recognition system
2. 3D and 2D feature toolkits
3. Gesture recognition
4. Object detection
5. Human computer interaction
6. Motion tracking

Dlib is initially a C++ library, but however Dlib does have a python API. Where all the functions of the library are also available when programming in python.

|  |  |
| --- | --- |
| Python | C# |
| Data types did not have to be an issue with python since it could be dynamically changed depending on how it is used | Data types had to be declared and the proper data type provided |
| Ran the program more smoothly compared to C# which resulted in less lagging | The smoothness of the program was not as well as python |
| Easy to load haar cascades as long as the haar cascade xml files were on directory it could be found | Difficult to load haar cascades since if not proper path provided led to many errors |
| Was quite easy to load libraries | Easier to load libraries but had issues with loading Dlib which is essential in calculating eye blinks |
| IDE USED – Spyder | |

## Chapter 4.2 Detecting face, eyes and smile

To detect the face and eyes of a user in front of a web camera, haar cascades classifier in the form of xml files are used to achieve this. Two steps are required in order to do this.

1. Firstly the classifier should be loaded



Figure 14 Code snippet

1. The loaded classifier can now be used

Figure 15Code snippet



The vital aspect of the detecting faces using the haar classifier are the parameters involved in the step 2 of the above context.

1. GrayFrame – mentioned in step 2 of the above context displays that the feed from the web camera should be converted to gray scale and set as the parameter according to (Krakowska, 2012)
2. Scale Factor – The image size or the frame which is grabbed from a series of frames from the web camera is reduced at every pass OpenCV makes over the frame or image. Increasing this value will make the face detector provide rapid results in detecting the face. If the passes made are too high the detector will be unsuccessful in detecting faces accurately. The normal value of this parameter is set to 1.1. It can increase to a range from 1.1 to 1.4 according to (Krakowska, 2012)
3. Minimum neighbor threshold – This value controls the raw detections of a face detected. If the value is set to 0. Then all the raw detections will be displayed of the user. When the value is higher the detection of the face is much stricter. In regard to when it is lower the detection is very lenient of the raw detections according to (Krakowska, 2012).

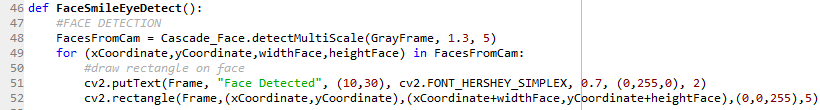


Figure 16Code snippet

This is the Face Smile Eye Detect method. When the face is detected. 4 parameters are returned which are the x coordinate, y coordinate and the width and the height of the user’s face. The x and y coordinate is located at the left upper corner of the user, to detect the whole face and draw a rectangle over the face we need the current “Frame”, X coordinate and Y coordinate. X coordinate + width and Y coordinate + height of the face. The colour in BGR form and the thickness of the line this is mentioned in line 52. Line 51 simply shows the user on the screen that the face is detected.

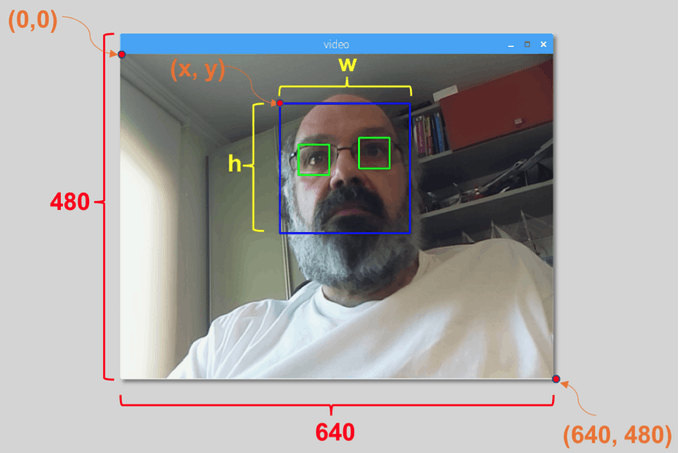


Figure 17(Medium, 2019)



Figure 18Code snippet

After we have identified the face of the user. We do not need to re detect the whole frame. We can limit the search area to the face area instead. Therefore we require a grayscale version of the user’s facial region. This is mentioned in line 54. We edit the grayscale frame we currently have. Similarly we do the same to line 55 in order to detect and identify the eyes and smile on the user’s face.



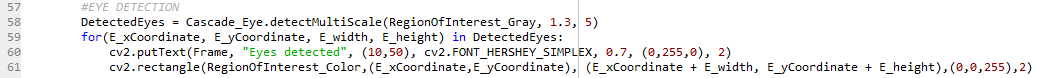
The eye haar cascade is loaded and ready for use in line 8 also the smile cascade is loaded and ready to user in line 9

Figure 19code snippet

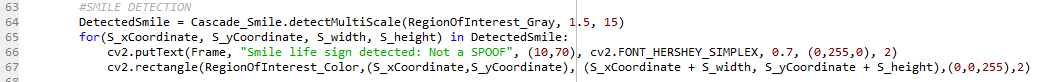
Similar to detecting the face of the user. The eyes can be detected using the haar cascade and the detect Multi scale method in line 58. When the eyes are also detected similar to detecting the face the 4 parameters are returned in line 59 by which the coordinates can be used in line 61 to draw a rectangle over the two eyes.

Figure 20Code snippet

Similar to detecting the face and eye of the user the haar cascade for smile can be used in line 64 to detect the smile from the trained dataset. Similarly the 4 parameters are returned. Where the user’s smile is detected and a rectangle drawn over it in line 67. The user’s smile detection to the user that the smile is detected in line 66.

## Chapter 4.3 Detecting facial landmarks

* One important function used from the Dlib is displayed below

Figure 21Code snippet



This function gets the frontal face detector and can be assigned to a variable. There are certain parameters required to get face detector to detect faces.

Figure 22Code snippet

1. GrayFrame - A grayscale image or frame from the web camera feed which is originally in color
2. 0 – The second parameter controls the delay of the capturing of frames, the default should be set to 0. If the value increases the frames grabbed from the web camera will be slower. Making it less efficient

* Another important function is the

Figure 23Code snippet

Once the faces are detected using the face detector from Dlib. The shape predictor can be used to detect facial landmarks. After the faces are detected



Figure 24Code snippet

The Face Predictor variable which was assigned the face landmark detector also known as the shape predictor from the Dlib. Requires two parameters.

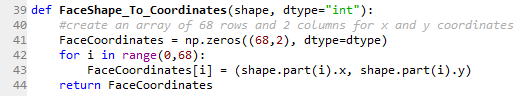
1. GrayFrame – The current gray scale images or frames from the web camera feed
2. face – which means the face detected from the face detector

Next the returned face shape is converted to a NumPy array.

* A NumPy array is also known as numerical python which is a library, available for python in order to interact with multi-dimensional arrays

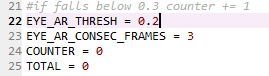
The NumPy array in this case should have to deal with 68 coordinates of a user’s face. Each x coordinate and y coordinate will be in this NumPy array.

A for loop can be used to loop through each x and y coordinate to draw points over the user’s facial landmarks.



The above method is required to convert the coordinates from the landmark detector into a NumPy array. This method initializes the array of 68 rows and 2 columns as mentioned in line 41 then returns the array in line 44

## Chapter 4.4 Detect eye blink



Line 22 is the EYE Threshold value. Which is the minimum value the eye aspect ratio can fall to in order to identify an eye blink

Line 23 is the number of frames the eye aspect ratio can stay in the threshold value for

Line 24 is the counter which counts the number of frames

Line 25 is the total number of blinks counter

To detect eye blink we need an index to where the eye is located in the face.



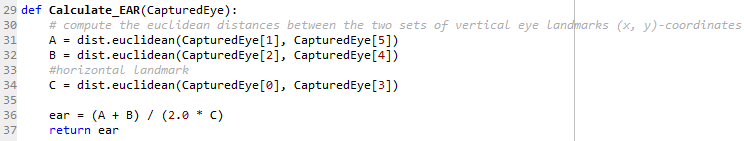
Firstly we get the indexes from face utils package which is a subsidiary package of the imutils package. The required indexes are provided when the key is provided like “left eye”.



Now we have the face shape and the coordinates of the face. The coordinates of the indexes of the array where the left eye and right eye is located are provided. For example: leftEye is having coordinates at index of array 40 to 52. At these indexes the eye coordinates can be found.



Now we calculate the eye aspect ratio for each eye. Each eye coordinates are sent to the Calculate\_EAR function



This is the Calculate\_EAR function. Here each eye coordinates are received and at line 31 and 32 the distance of the horizontal length is calculated. In line 34 the distance between the two points on the either side of the eye is calculated. To calculate the eye aspect ratio. In line 36. Vertical distance is added and divided by the horizontal distance multiplied by 2. This value is returned.



The value of eye aspect ratio of each eye is now calculate and averaged.



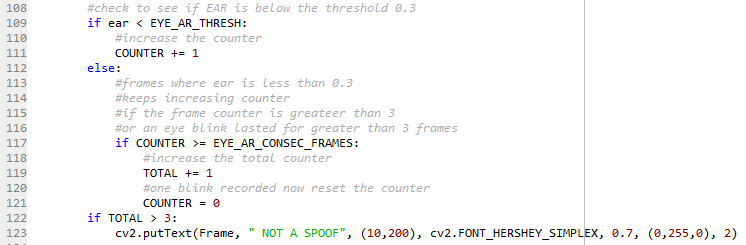
In order to show the user we first draw a convex hull over each eye. A convex has no interior angles. The left eye provided all the coordinates to draw the convex hull. The hull is a sort of a shell over each eye. Similar thing is done to the right eye in line



After the convex hull is drawn over the eye the contours can be drawn around the eye to show the eye boundary. This is done for each eye in line 102 and 103.



The user is at first regarded as a spoof until proven otherwise with life signs. Therefore the total number of blinks is checked in line 105 and then displayed as spoof for user in line 106.



First we check if the eye aspect ratio is less than the threshold value. If it’s so we increase the counter by 1 at line 111. Else if the counter which counts the frames have risen of equalled to the EYE\_AR\_CONSEC\_FRAMES value in line 117. This is regarded as an eye blink then the total counter is increased by 1 in line 119. Then counter is reset to 0 in line 121. If the total eye blinks has risen above 3 the user is not regarded as a spoof anymore then the user is advised as such using the line 123.

# Chapter 5. Testing

The program needs to be tested in many different ways. First and foremost a test plan and test criteria must be determined. According to the test criteria the tests can be carried out.

|  |  |  |
| --- | --- | --- |
| Test 1 | Pass/Fail | Remarks |
| Face detection and eye detection | PASS | Face was detected successfully, also the eyes were detected successfully and displayed to the user that these features were detected |
| Figure 25Test 1 snapshot | | |

|  |  |  |
| --- | --- | --- |
| Test 2 | Pass/Fail | Remarks |
| Smile detection | PASS | After face and eyes were detected the smile was also detected showing life signs |
| Figure 26 Test 2 snapshot | | |

|  |  |  |
| --- | --- | --- |
| Test 3 | Pass/Fail | Remarks |
| Spoof detection | FAIL | Face features could not be detected under very dark background lighting. Only the face was detected |
| Figure 27Test 3 snapshot | | |

|  |  |  |
| --- | --- | --- |
| Test 4 | Pass/Fail | Remarks |
| Spoof detection | FAIL | Eye and face features were detected with low light, but depending on the web camera feed the spoof detection went haywire. Even when not smiling the smile was detected because of bad lighting and camera quality |
| Figure 28Test 4 snapshot | | |

|  |  |  |
| --- | --- | --- |
| Test 5 | Pass/Fail | Remarks |
| Eye blink, eye aspect ratio counter test | PASS | Eye blink counter increased at every eye blink. Eye aspect ratio counter is also working. When eye aspect ratio falls to zero the eye blink counter is increased by 1 |
| Figure 29Test 5 snapshot | | |

|  |  |  |
| --- | --- | --- |
| Test 6 | Pass/Fail | Remarks |
| Eye blink, eye aspect ratio counter test with face movement to the side | PASS | Blinks were detected and eye aspect ratio was working, but raw detections were found. |
| Figure 30Test 6 snapshot | | |

|  |  |  |
| --- | --- | --- |
| Test 7 | Pass/Fail | Remarks |
| Detect spoof | PASS | User is recognized as a spoof until the user shows liveness evidence |
| Figure 31Test 7 snapshot | | |

|  |  |  |
| --- | --- | --- |
| Test 8 | Pass/Fail | Remarks |
| Detect liveness | PASS | After a certain period of blink checking the user regarded as a spoof is not regarded as a spoof anymore. |
| Figure 32Test 8 snapshot | | |

# Chapter 6. Critical Appraisal

There was a lot of research involved in regard to this report. The main goals were

1. Detect face
2. Detect eyes
3. Detect eye blink
4. Detect spoof

Initially the project started out with C# and EmguCV which is a wrapper class for OpenCV. The Face detection and eye detection were quite easy to implement. Since haar cascades were involved and many tutorials were helpful.

When it came to detecting the eye blink, it was essential that a facial landmark detection was absolutely essential in completing this task. Sadly EmguCV was not able to provide help in regard to this area. Therefore Dlib was available which allowed to detect facial landmarks quite easily, but since it was not available for C# the program had to be switched to python language. The syntax of the programming language change was quite easy to understand.

With the help of Dlib and many libraries of python the eye blink detection was a success. Since these 3 goals were successfully reached the Spoof detection was easy to implement.

|  |  |
| --- | --- |
| Goal | Remarks |
| Detect face eyes and smile | After successfully detecting the face and eyes. Even smile detection was implemented |
| Detect eye blink | Successfully able to detect eye blink |
| Detect spoof | Spoof detection is working best on a mid-high end computer with proper lighting |

## Chapter 6.1 what did I learn?

* Computer vision
* Haar cascades
* Open CV and wrapper Emgu CV
* Landmark detections
* Eye Blink detection
* Eye face and smile detection

I learned a lot about computer vision. How computers identify objects and what algorithms are used in computer vision. I learned about haar cascades and how they work in identifying human faces. I learned about Open CV which was very helpful in completing the objectives set at the start of the project. I learned about python and Emgu CV which is a wrapper of Open CV even though the wrapper did have certain limitations. I learned about facial landmarks and how to detect these landmarks, by which I gained the knowledge of learning about the eye aspect ratio, which was really helpful in detecting liveness using eye blink. I learned how to detect a face, and then use this facial region as a scope to detect other secondary features such as eyes and smile.

## Chapter 6.2 Limitations

* Since this system is implemented with a web camera and since everyone has one. It is easily accessible and easy to use. The disadvantage of using such a device is that the quality of data cannot be regarded as high level data, since most web cameras provide a moderate level data quality.
* The system requires proper lighting therefore will not work well under bad lighting conditions
* If the user keeps moving or keeps a hand over his eyes or sunglasses were worn during the spoof detection phase it will not work

## Chapter 6.3 Overcoming limitations

* Could put to use better camera equipment to get a better data quality
* Need proper lighting
* The user needs to not wear any sunglasses, and should stay still when system is detecting the spoof

## Chapter 6.4 Commercial product?

Spoof detection is a necessity these days. Many authentication and authorization based security companies are looking to add certain liveness detection measures to minimize spoofing. This application of eye blink detection and multiple facial area detections also known as liveness detection can also be used in different applications such as

1. Face recognition
2. Driver drowsiness detection
3. Communication method between a disabled person using blinking

## Chapter 6.5 Weaknesses and how to overcome them!

* Encapsulation, polymorphism and inheritance could be used

This system even though written in python, since it’s an object oriented language. More OOP concepts can be used to secure data and reuse this python code in a better way.

* Texture detection is possible

Texture detection could also be implemented.

# References

1. Office for Science, G. (2018) ‘Biometrics: a guide i’. Available at: <https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/715925/biometrics_final.pdf>.
2. Bolle, R. (2011). *Guide to biometrics*. New York: Springer.
3. Pan, G., Wu, Z. and Sun, L., 2008. Liveness detection for face recognition. In *Recent advances in face recognition*. IntechOpen.
4. Ross, A.; Nandakumar, K. & Jain, A.K. (2006). Handbook of Multibiometrics, Springer Verlag.
5. Bradski, G. and Kaehler, A. (2012). *Learning OpenCV*. Beijing: O'Reilly.
6. Junaidy, D., Wulandari, M. and Tanudjaja, H. (2019). Real time face detection using haar-like feature method and local binary pattern method. *IOP Conference Series: Materials Science and Engineering*, 508, p.012076.
7. Cech, J. (2016) ‘Real-Time Eye Blink Detection using Facial Landmarks’.
8. Krakowska, P. *et al.* (2012) ‘FACE DETECTION IN C #’.
9. Computer Vision (2018) YouTube Video, added by Balazs Holczer [Online]. Available at <https://www.youtube.com/watch?v=F5rysk51txQ> [Accessed 25 June 2019]
10. Rosebrock, A. (2019). *Eye blink detection with OpenCV, Python, and dlib - PyImageSearch*. [online] PyImageSearch. Available at: https://www.pyimagesearch.com/2017/04/24/eye-blink-detection-opencv-python-dlib/ [Accessed 22 Jul. 2019].
11. Medium. (2019). *Real-Time Face Recognition: An End-To-End Project*. [online] Available at: https://towardsdatascience.com/real-time-face-recognition-an-end-to-end-project-b738bb0f7348 [Accessed 22 Jul. 2019].