Anti-Spoofing Techniques in Face Recognition, an Ensemble Based Approach

Răzvan-Daniel ALBU[1], Cornelia Emilia GORDAN[1], Ioan DZIȚAC[2,3]*

[1] University of Oradea, Faculty of Electrical Engineering and Information Technology,

Department of Electronics and Telecommunications, 1 Universității, 410087, Oradea, Romania
ralbu@uoradea.ro

[2] Aurel Vlaicu University of Arad, Faculty of Exact Sciences,

Department of Mathematics and Informatics, 2 Elena Drăgoi Street, 310330 Arad, Romania
ioan.dzitac@uav.ro (**Corresponding author*)

[3] Agora University of Oradea, Faculty of Economics, Department of Economics,

8 Piața Tineretului, 410526 Oradea, Romania

idzitac@univagora.ro

Abstract: In this article we describe the implementation of a reliable and innovative ensemble-based technique that can prevent face spoofing attacks. The presented software is part of a technology developed in partnership with IsItYou, an Israeli company, that attempts to replace passwords with a face-based authentication system. Since the main problem of biometric systems is represented by the spoof attacks, IsItYou came with a solution to this, developing a unique technology that can identify spoof attacks, and authenticate only authorized humans. Inspired from deep learning techniques where ensemble-based solutions improve machine learning results by uniting several models, a software ensemble that combines multiple anti-spoofing methods, covering a larger range of spoof attacks and increasing overall security was developed. The article also shows the performances results and implementation details. The experimental results signpost our solution can provide first-rate results compared to the state-of-the-art approaches.

Keywords: Face anti-spoofing, Face recognition, Beware, Biometric security, Ensemble system.

1. Introduction

In this research work a technology developed together with IsItYou, an Israeli company, is presented (ISITYOU, 2018). The main objective of IsItYou is to replace passwords with a reliable and safe face recognition solution that works flawlessly without any user intervention. Basically, the user just looks at his device, its camera looks back at him, and can recognize him, and say if he is a real human or not. As said by Acuity Market Intelligence, the mobile biometric market will theoretically blast from $1.6 billion in 2014 to $34.6 billion in 2020 (AQUITY-MI, 2019). Our research is well in line with the statement of IsItYou company which foresees we'll be able to log in our bank account, rent a car, open doors, sign legal documents, and do everything else just by showing our unique face, in a not-so-distant future. However, like most new technologies it produces new vulnerabilities. And one of the most popular ways to cheat a face recognition software, is a 'face spoof' attack. A spoofing attack is an attempt to obtain someone else's access rights by using a photo, video or a different substitute for an authorized face. So, a unique technology that has anti-spoofing capabilities was developed. In order to increase the security, it is a common practice of the usage of biometric applications to come in pairs: fingerprint and iris, fingerprint and voice, face and voice etc. The technology presented in this paper needs no additional biometrics as a 2nd factor, since the anti-spoof capabilities act as 2nd factor thus increasing security and reducing deployment cost (KAIROS, 2018). This technology is based on OpenCV, but as it can be seen in (OPENCV, 2019) OpenCV has no face recognition capabilities. OpenCV (Open Source Computer Vision Library) is an open source project, that offers computer vision and machine learning functions, as a library. OpenCV was created to offer a shared set-up for computer vision projects and to hasten the integration of machine perception capabilities in the saleable software products. Also, this technology adds to face detection and multi face tracking capabilities offered by OpenCV, a face recognition algorithm, and anti-spoofing capabilities. Anti-spoofing algorithms are the major differentiator between the face-base authentication service proposed in this research and any other existing similar service. The idea behind ensemble-based systems is to benefit from all the advantages offered by different techniques and methods that act as subsystems and solve different parts of a complex problem. The core of this technology is written in C++ and consists in two big parts: face recognition and face anti-spoofing.

In this article the innovative face anti-spoofing solution which was built by combining multiple types of attacks detectors into an ensemble with high accuracy is described. The proposed ensemble is also a BEWARE (Biometric Early Warning Rating Engine), consumed by various client applications that run on all major platforms (Windows, Linux, android, IOS).

This paper is structured as follows:

- First chapter presents the main subject, the problem we try to solve and why it is important.
- The second chapter presents similar state-of-the-art research works and a brief specialized literature review.
- In the third chapter the main architecture of the proposed anti-spoofing solution is described, and relevant implementation details are provided. Each module of the proposed ensemble is described in a separate sub-chapter.
- Next, experiments and performance results are provided, and the paper ends with relevant conclusions and future working plans.

2. Literature Review

Spoofing is the main vulnerability of a biometric system. It is defined as the process where someone attempts to authenticate using fake biometric samples. Because face is currently the most accessible biometric system, there have been numerous diverse types of spoof attacks for faces including print attack, replay attack, 3D masks attack, etc. (Erdogmus & Marcel, 2013) Analysing the specialised literature, it can be noticed numerous anti-spoofing techniques were proposed, including methods based on texture analysis, contextual information techniques, life signs detection, motion analysis, colour analysis, or deep learning-based methods (Galbally et al., 2014), (Varghese & Mathew, 2015). In (Jourabloo et al., 2018) authors try to implement face anti-spoofing solution by reversely decomposing a fake face into the real face and the spoof noise pattern. In (Bharadwaj et al., 2013) there was an attempt to identify spoof attacks by using a texture analysis approach based on Local Binary Patterns (LBP) on multiple types of attacks (with printed copies of faces, or with photos and videos displayed on devices of different sizes). There are also hardware-based methods that incorporate extra devices, like 3D cameras, in order to identify spoof attempts. "Erdogmus & Marcel (2018)" made use of depth maps to distinguish between the real 3D faces and the 2D fake faces. The motion detection methods have also been studied very carefully since 2007, and they were also applied in the research presented in this paper, but eye blinking detection (Pan et. al, 2007) was found to be not very useful in real life scenarios since it requires relatively long time to accumulate necessary information, and it can also be spoofed without difficulty. Similar and interesting research works that use motion magnification, image distortion analysis or liveness detection (Pereira et al., 2014), can also be found in (Zhang, et al., 2012).

3. Main Architecture

The main architecture of the face recognition technology proposed in this paper is described in Figure 1. First, an image is captured, and face detection is invoked. If face detection fails, there is no point to continue, and the access is denied. In the C++ core of this technology, the face detection function was implemented by making use of OpenCV implementation of the Viola-Jones algorithm. If a face can be detected, the next step is to see if the user is recognized or not, and it represents the stage where the image is input into the proposed face recognition module. This module will create a biometric template for the input image and will match it against all biometric templates of all enrolled users in the system, providing a matching score for each one. Based on those scores it will be decided if the user is recognized or not as being one of the enrolled users. If the user is not recognized the access is denied. If the user is recognized, the anti-spoofing ensemble is invoked, trying to verify if this is a spoof attempt or a legitimate authentication. The anti-spoofing ensemble architecture is illustrated

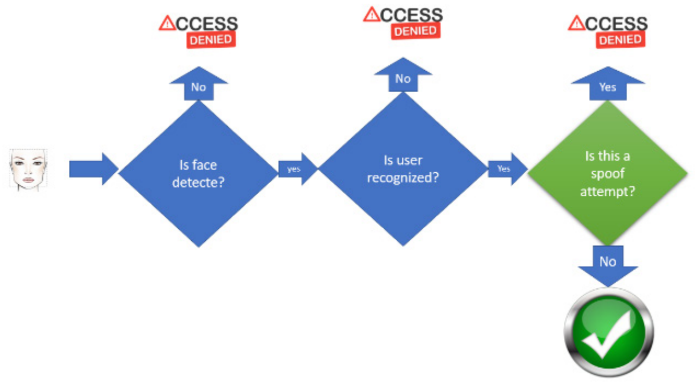


Figure 1. Authentication process flow

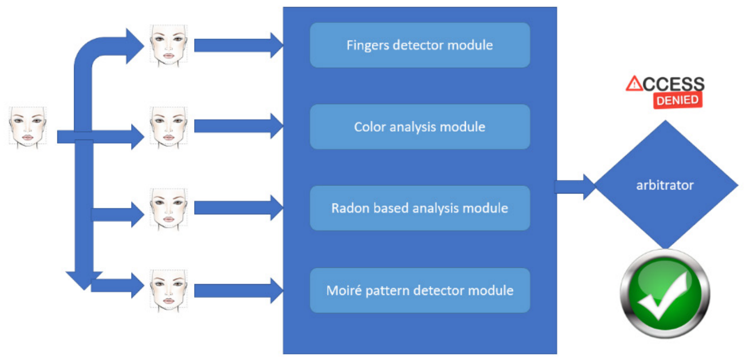


Figure 2. Ensemble architecture

in Figure 2. As it can be seen the input image is cloned and input in parallel to all four anti-spoofing modules that run in different threads. Each module will return a binary decision, which will be true if it is a spoof attempt. All the decisions will be processed by the arbitrator module which will make the final decision. If at least one module finds a spoof attempt the final decision will be access denied. If none of modules finds a spoof attempt the input image is considered real and the access is granted.

3.1 Fingers Detector Module

A very simple and often performed spoof attack is achieved by numerous users by taking a picture of someone on their phone, or on a printed paper, and then trying to hold in hand the phone that displays that image of that person, in front of the camera of the authentication system. Thus, a method for detecting such attacks (see Figure 3) was developed by using a Haar-cascade Classifier.

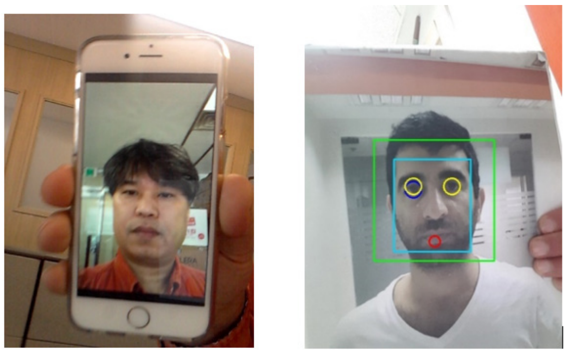
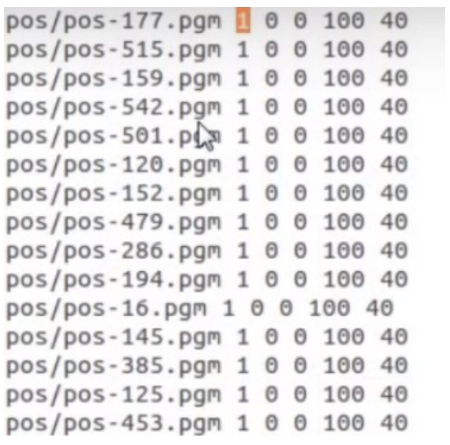


Figure 3. Fingers on the edge

Neighbouring rectangular regions of a well-defined location are considered in an election window. Next the pixel intensities in each region are totalized and the difference between these sums is computed. In order to detect the object a window is moved over the image, and at every location each stage of the classifier labels that region as being either positive or negative. Positive means that an object was found while negative means that the specified object was not found in that window. If the current region was labelled as negative, then the window is moved to the next location. If the labelling returns a positive result, then the region moves off to the next stage of classification. The classifier produces a final decision which is positive, when all the stages returned a result, indicating that the object was found in the image.

To train our classifier for fingers detection Linux Ubuntu 12.04.5 LTS and OpenCV 3.0 were used. The first step was to gather a database consisting of more than 1000 images of this type of attack. Those images were grouped in two folders: positive images and negative images. The positive images are images which contain a face on a screen of a phone held in front of the camera with the fingers. The negative images are images of real faces, which do not contain fingers that hold a screen displaying a face. The positive and negatives images sets have similar sizes, around 5000 images each. All the images have the size of 100x40 pixels. For the positive images an info file was created where the location of fingers in each image was described. This file is shown in Figure 4.



```
pos/pos-177.pgm 1 0 0 100 40
pos/pos-515.pgm 1 0 0 100 40
pos/pos-159.pgm 1 0 0 100 40
pos/pos-542.pgm 1 0 0 100 40
pos/pos-501.pgm 1 0 0 100 40
pos/pos-120.pgm 1 0 0 100 40
pos/pos-152.pgm 1 0 0 100 40
pos/pos-479.pgm 1 0 0 100 40
pos/pos-286.pgm 1 0 0 100 40
pos/pos-194.pgm 1 0 0 100 40
pos/pos-16.pgm 1 0 0 100 40
pos/pos-145.pgm 1 0 0 100 40
pos/pos-385.pgm 1 0 0 100 40
pos/pos-125.pgm 1 0 0 100 40
pos/pos-453.pgm 1 0 0 100 40
```

Figure 4. Fingers.info file

The first value, which is 1, means that each image contains only one object that should be detected. The next four values are the coordinates of the object of interest present in the image. As it can be noticed, the coordinates indicate that the full image represents the object, because it was previously cropped manually from the original images, when the sets of positive and negative images were created. Since the images size can dramatically influence the training time, only the regions of interest need to be cropped from the full-size images. A similar file is also required for the negative images, but this is simpler since it contains only the list of images names from the folder with negative images. Next, we created the vector (VEC) file using OpenCV framework calling the create Sample utility with the following command:

```
$opencv_createsample -info fingers.info -num
550 -w 48 -h 24 -vec fingers.vec
```

The next step is to train the cascade using this vector file, as follows:

```
$opencv_traincascade -data data -vec fingers.
vec -bg bg.txt -numPos 550 -numNeg 450 -num-
Stages 20 -w 48 -h 24 -featuretype HAAR.
```

Data is the empty directory where all files will be stored. -vec argument specifies the utilized vector file, while bg.txt contains the background images which will be used in order to train the cascade. The number of positive images should be lower than the maximum number of vector images. The number of stages specify how many stags will be included in the Viola Jones algorithm. The higher the better but the training time will also increase. The features type is set to HAAR by default. This will provide better results, but the training can take weeks: when the training was finished, the fingers cascade.xml file was obtained. This file can be used in the C++ code present in this paper in order to detect fingers on the edge of a phone which displays a face, in an input image, as follows:

```
CascadeClassifier fingers_cascade;
    if( ! fingers_cascade.load( 'fingers_cas-
cade.xml'  )  ){  printf("--(!)Error  loading
cherries cascade\n"); return -1; };
cv::Mat frame_gray;
std::vector<Rect> fingers;
cvtColor( inputimage, inputimage_gray, COL-
OR_BGR2GRAY );
equalizeHist(  inputimage_gray,  inputimage_
gray );
fingers_cascade.detectMultiScale(   inputim-
age_gray, fingers, 1.1, 2, 0|CASCADE_SCALE_
IMAGE, Size(50, 100) );
for ( size_t i = 0; i < fingers.size(); i++ )
    {
      Point center( fingrs[i].x + fingers[i].
width/2, fingers[i].y + fingers[i].height/2 );
        ellipse( inputimage, center, Size(
fingers[i].width/2, fingers[i].height/2 ), 0,
0, 360, Scalar( 255, 0, 255 ), 3, 4, 0 );
        Mat faceROI = inputimage_gray( fin-
gers[i] );
```

The attacker may try to avoid this test by putting the image closer to the camera, so that the edges and fingers won't be visible. To avoid this scenario, the size of the detected face was limited. If the face is too big, the face will not be detected, and without a face, none of the spoof checks would be invoked, and the input image would be automatically rejected. The user was also instructed to move further from the camera if he was too close to it.

3.2 Colour Analysis Module

This module tries to detect spoof attacks based on the idea that a picture of a screen on which is displayed a face, has a higher blue light component which can be emphasized (Figure 5). Here, certain parameters for an input image, such as luminosity, contrast, blue flag, red flag, green flag are computed. and try to decide if the image is real or fake, based on a complex decision algorithm.

```
testRGB_11    =  std2(handles.ColorImag1(:,:,1));
testRGB_21    =  std2(handles.ColorImag1(:,:,2));
testRGB_31    =  std2(handles.ColorImag1(:,:,3));
meanR1   =   mean2(handles.ColorImag1(:,:,1));
meanG1   =   mean2(handles.ColorImag1(:,:,2));
meanB1   =   mean2(handles.ColorImag1(:,:,3));
B_ratio_value1   =    meanB1 / ((meanR1 + meanG1)
/2);
R_ratio_value1   =    meanR1 / ((meanB1 + meanG1)
/2);
%% Compute the Contrast, Luminance and Ratio value
ContrastRef        =        mean([mean(testRGB_11),
mean(testRGB_21), mean(testRGB_31)]);
LumRef = mean([meanR1, meanG1, meanB1]);
Ratio_RB1   =   R_ratio_value1/B_ratio_value1;
```

The final decision algorithm is very complex, and it can represent itself the subject of a different article. Briefly, this method focuses on quantifying how much artificial blue is on a face, relative to red and green components of the image, and on verifying if luminosity is increased due to an additional light source (the screens on which it was displayed) and if the contrast is affected or not. Based on the experiments carried out it was noticed that fake images tend to have a higher contrast and luminosity than the real images and a greater blue component.

The phrase "fake image" refers to spoof attempts, the image of an image, while real images are images of the authorized human. By contrast we understand here the Root mean square (RMS) contrast (1), for indoor images, defined as:

$$\sqrt{\frac{1}{MN}\sum_i \sum_j (I_{ij} - \vec{I})^2} \quad (1)$$

where intensities I_{ij} are the i-th, j-th elements of the image having a size of M x N pixels, while $\bar{I}$ is the average intensity computed using all the pixel values.

For outdoor images, the Weber contrast (2) was used (2) for images where small features were present on a large uniform background, and Michelson contrast (3) for images where both bright and dark features occupied comparable fractions of the area.

$$\frac{I - I_B}{I_B} \quad (2)$$

where I signifies the luminance of the features, and I_B stands for the luminance of the background.

$$\frac{I_{max} - I_{min}}{I_{max} + I_{min}} \quad (3)$$

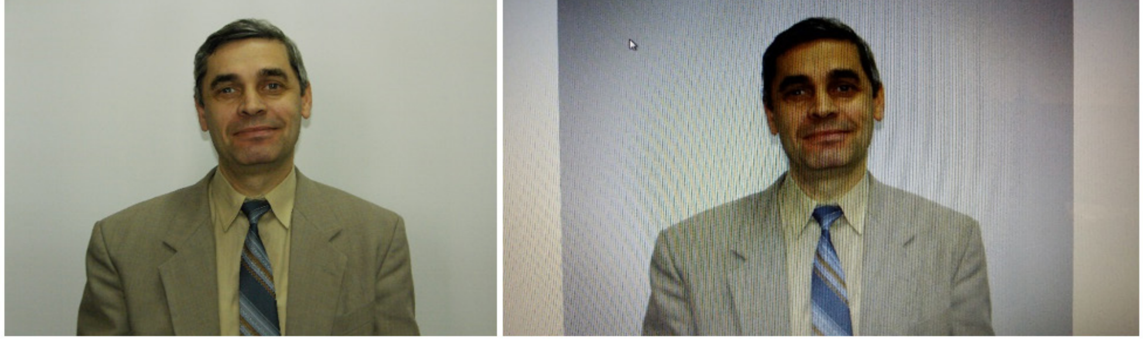


Figure 5. Real image versus Fake image (Moire patterns and abnormal blue)

3.3 Radon Based Analysis Module

This module was described in detail in a different article (e.g. Albu, 2015). The goal of this anti-spoofing method is to create a unique signature for each image, using Radon transform. This signature will contain information about both basic image parameters, such as luminosity or contrast, and its shape. For a given function f, the Radon transform of f is defined, for each pair of real numbers (t,θ), by:

$$Rf(t,\theta) = \int_{s=-\infty}^{\infty} f(t*cos(\theta) - s*\sin(\theta), t*\sin(\theta) + s*\cos(\theta))ds \quad (4)$$

This method is based on the idea that attack images manage to deliver smoother and sharper Radon based signatures than real images. it can also be noticed that the signatures for fake and real images have different amplitudes, shapes and number of spikes. The algorithm used to generate this digital signature works as follows.

- Read the input image, which represents the face cropped form original image
- Resize it to 100x100 pixels
- Initialize n angles vector, theta, to take values from 0 to 180
- Perform the Radon transform of the input image for all the angles in theta
- The results will be a matrix having on each column the Radon transform of the given image at a theta angle
- Generate the digital signature by extracting the maximum values of each column of this matrix

3.4 Moiré Pattern Detector Module

When a pattern on a photographed object interferes with the shape of the light sensors, it can generate unwanted artefacts named Moiré patterns. An example of Moiré pattern can be observed in (Figure 5). This module is meant to detect the Moiré patterns by using peak detection in the frequency domain and to use them to identify spoof attempts. The algorithm presented in this paper implies that real images comprise most of their energy at low frequencies, while fake images have unusual peaks at higher frequencies (Crowley & Stern, 1984). The main objective is to identify peaks at frequencies different than the baseband (Garcia & de Queiroz, 2015). For an input image of a face, diverse band-pass-filtered versions of this image are generated, and a peak detector is applied to the absolute value of the DFT (Discrete Fourier Transform) of each of these filtered versions. If any strong peak is detected, the image is considered a fake image (a spoof attack). Each band-pass-filtered version of I(IBP) is obtained through the convolution of I with a difference-of-Gaussians (DoG) filter (Wen et al., 2015). In order to avoid border-continuity artefacts, a Henning window was applied prior to the calculation of the DFTs.

4. Experiments and Results

The IsItYou ensemble was tested with a large database of fake images of 25 individuals (IsItYou members and family) under variable conditions, for a total of 500 images. The aforementioned images were photographed under several conditions: (i) displayed on a 19-inch Asus G750jx laptop screen and captured by an Note 3 camera; (ii) displayed on a13-inch Macbook Pro screen and captured by an iPhone 4 camera; (iii) displayed on an Samsung smart TV screen and captured by an iPhone 4 camera; and (iv) displayed on an iPhone 4 screen and captured by an Note 3 camera.

In all these conditions, images were taken at different distances from the displays, in order to assess how IsItYou ensemble operates under these circumstances. Image resolution was always constant at 640x480 pixels. Table 1 summarizes the performed experiments and shows the obtained results in terms of accuracy (100%-FalseAccepts%-FalseRejects%).

5. Conclusion

This paper presents an ensembled-based solution for face anti-spoofing. This ensemble consists in multiple anti-spoofing methods, each one being explained and described. The results prove its efficiency and capability to detect most common spoof attempts. The presented solution can add a layer of security to any face recognition system. In contrast with other anti-spoofing attempts, the technology present in this research needs no additional biometrics as a 2nd factor, thus increasing security and reducing deployment cost. It is a cross-platform software, has high accuracy, and even if it is not 100% accurate, it is harder

Table 1. Ensemble accuracy

Condition	Display	Capture	Distance	Accuracy
1	19-inch Asus G750jx	Note 3	15cm	93.3%
2	19-inch Asus G750jx	Note 3	20cm	94.3%
3	19-inch Asus G750jx	Note 3	25cm	93.8%
4	19-inch Asus G750jx	Note 3	30cm	94.1%
5	19-inch Asus G750jx	Note 3	35cm	92.8%
6	a13-inch Macbook Pro	iPhone 4	15cm	96.6%
7	a13-inch Macbook Pro	iPhone 4	20cm	97.3%
8	a13-inch Macbook Pro	iPhone 4	25cm	95.5%
9	a13-inch Macbook Pro	iPhone 4	30cm	95.2%
10	a13-inch Macbook Pro	iPhone 4	35cm	96.1%
11	Samsung Smart TV	iPhone 4	15cm	90.3%
12	Samsung Smart TV	iPhone 4	20cm	91.2%
13	Samsung Smart TV	iPhone 4	25cm	90.4%
14	Samsung Smart TV	iPhone 4	30cm	91.6%
15	Samsung Smart TV	iPhone 4	35cm	92.3%
16	iPhone 4	Note 3	15cm	95.5%
17	iPhone 4	Note 3	20cm	96.3%
18	iPhone 4	Note 3	25cm	94.9%
19	iPhone 4	Note 3	30cm	97.6%
20	iPhone 4	Note 3	35cm	98.1%

to be hacked or fooled than a password-based authentication system. Theoretically, password-based system has higher accuracy, but in real life, very long and complicated passwords which are impossible to be hacked are also hard to remember. These passwords come up with a new vulnerability requiring a safe place to be stored. On the other hand, easy to remember passwords are useless and easy to detect by hackers, offering a very low actual security. A face-based authentication system is a very convenient solution, because the user just shows up in front of a camera and is recognized. If it can also be made secure, we think it may replace passwords. As future working plans, new anti-spoofing modules may be added to this ensemble, at least one for liveness detection, which is currently under development, and one based on deep learning.

Acknowledgements

This research is partially supported by “Cercetare Dezvoltare Agora” (R & D Agora) of Agora University from Oradea, Romania.

REFERENCES

1. Albu, R. D. (2015). Face Anti-Spoofing Based on Radon Transform. In *The 13th International Conference on Engineering of Modern Electric Systems*, Oradea, Romania (pp. 124-127), DOI: 10.1109/EMES.2015.7158435

2. AQUITY-MI, <http://www.acuity-mi.com/>, [January 2019].

3. Bharadwaj, S., Dhamecha, T. I., Vatsa, M. & Singh, R. (2013). Computationally efficient face spoofing detection with motion magnification. *In IEEE, International Conference on Computer Vision and Pattern Recognition Workshops* (pp. 105-110).

4. Crowley, J. & Stern, R. (1984). Fast computation of the difference of low-pass transform, *IEEE Transactions on Pattern Analysis*, *PAMI-6*(2), 212–222.

5. Erdogmus, N. & Marcel, S. (2013). Spoofing attacks to 2D face recognition systems with 3D masks. In *International Conference of the Biometrics Special Interest Group*, (*EPFL-CONF-192407*) (pp. 184-189).

6. Galbally, J., Marcel, S. & Fierrez, J. (2014). Image quality assessment for fake biometric detection: application to iris, fingerprint, and face recognition, *IEEE Transactions on Image Processing*, *23*(2), 710-724.

7. Garcia, D. C. & de Queiroz, R. L. (2015). Face-Spoofing 2D-Detection Based on Moire-Pattern Analysis, *IEEE Transactions on Information Forensics and Security*, *10*(4), 778-786, ISSN: 1556-6013.

8. ISITYOU, <https://www.isityou.biz/> [January 2018].

9. Jourabloo, A., Liu, Y. & Liu, X. (2018). Face De-Spoofing: Anti-Spoofing via Noise Modeling. In *Proceedings of the European Conference on Computer Vision (ECCV)*, Munich, Germany (pp. 88-93), DOI:10.1007/978-3-030-01261-8_18

10. KAIROS, <https://www.kairos.com/blog/face-recognition-kairos-vs-microsoft-vs-google-vs-amazon-vs-opencv> [April 2018].

11. OPENCV, <https://opencv.org/about.html> [January 2019].

12. Pan, G., Sun, L., Wu, Z. et al. (2007). Eyeblink-based Anti-Spoofing in Face Recognition from a Generic Webcamera. In *IEEE International Conference on Computer Vision* (pp. 144-147).

13. Pereira, T., Komulainen, J., Anjos, A., De Martino, J. M., Hadid, A., Pietik¨ainen, M. & Marcel, S. (2014). Face liveness detection using dynamic texture, *EURASIP Journal of Image and Video Processing*, *2014(1):*2, 1-15.

14. Varghese, R. A. & Mathew, J. S. (2015). Face Anti-spoofing methods, *IJSTE - International Journal of Science Technology & Engineering*, *2*(4), 318-320, ISSN (online): 2349-784X.

15. Wen, D., Han, H. & Jain, A. K. (2015). Face Spoof Detection with Image Distortion Analysis, *IEEE Transactions on Information Forensics & Security*, *10*(4), 746-761, DOI: 10.1109/TIFS.2015.2400395

16. Zhang, Z., Yan, J., Liu, S., Lei, Z., Yi, D. & Li, S. Z. (2012). A face anti-spoofing database with diverse attacks. In *IEEE International Conference of the Biometrics* (pp. 26-31).