Using a Challenge to Improve Face Spoofing Detection

Jeroen Fokkema
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
j.fokkema@student.utwente.nl

ABSTRACT

Biometric face recognition is becoming a popular authentication method on i.a. mobile devices. With this application increasing, there comes a natural increase in interesting attack scenarios for people with bad intentions. One of the ways to trick biometric recognition methods is to spoof the system by showing an artificial reproduction of somebody else his face. This can be done by e.g. showing a printed picture, a video or by wearing a mask.
Multiple methods are developed to detect this kind of spoofing, but these are still lacking performance in terms of detection rates. Also, many of these methods are not applicable to mobile devices, due to specific constraints that exists on these devices.
In this paper we extend an existing spoofing detection method with usability on mobile devices in mind. The existing method uses 3D properties of the face to distinguish between a real face and 2D representations of a face. The extension on this method is a challenge, which requires the user to rotate its head in a specific pattern. By adding this challenge, the necessary rotation for the 3D check is enforced and furthermore, theoretically, more types of spoofing can be detected. Tests shows that users can successfully perform the challenge and that it is capable of reliably detecting spoofing attacks.

1. INTRODUCTION

Research in the field of face spoofing detection techniques has begun around 2004[19]. Around that time the application of biometric authentication was increasing. This caused a growing attack surface for attackers to spoof biometric recognition systems[1]. These attacks can be performed on many different components of a biometric system (see figure 1). Next to sensor attacks, which will be explained later, the system can be attacked in multiple ways from the inside. Communication can be intercepted and replaced with false footage or even with false authentication decisions. Databases can be accessed, giving the opportunity to alter biometric data. This allows people to be added to the database (to let unauthorized people access the system) or deleted from the database (to let authorized people be denied by the system). To protect against these kinds of attacks, the internal communication and storage of the biometric authentication system should be properly secured.

On the other hand, sensor level attacks are attacks that are specifically targeted at the sensor of the biometric system. This type of attack does not require access to the internal working of the system, but can be performed from the outside. In the case of a face recognition systems, these attacks consists mostly of spoofed faces that are presented to the camera. These faces can be spoofed using a picture of another person — printed on paper or displayed on a screen — a video or a 3D mask.

To boost research in the field of biometric spoofing, the Tabula Rasa project was started in 2011[2]. This research project provided a platform for more than 70 papers about face spoofing to be published, mostly in the field of detecting sensor-level attacks. Since then, the amount of spoofing detection methods and their performance has greatly improved. However, spoofing is still hard to detect and especially when real-life scenarios are tested – for example with cross-database testing – error rates are still above 10%.

Another problem is that most testing is done using static cameras, while biometric authentication gets increasingly popular on mobile devices. Many existing spoofing detection methods are difficult to port to mobile devices, because these devices pose some extra difficulties compared to static systems. Smith et al. [28, 29] and Wen et al. [32] identified the following difficulties for spoofing detection on mobile devices:

- Increased movement of the face compared to fixed cameras. This increases the difficulty for face detectors to accurately detect the face;
- Differing lighting conditions resulted in inconsistent behaviour of face detectors;
- The cameras on mobile devices all behave differently in terms of auto-focus, exposure and auto white-balance. Using manual settings could not resolve these difficulties;
- Hardware limitations cause a trade-off between resolution and frame-rate;
- Illumination levels are unpredictable;
- Phones do not capture atomic frames, but 'scan' the image;
- The cameras on phones often have a narrow dynamic range;
- Metering and auto-focus are sometimes inaccurate, resulting in over exposed or blurry footage.

Next to these, there are some other difficulties like the limited computational power of mobile phones and the fact

[1]For example, in 2009 the biometric verification on Lenovo, Asus and Toshiba face recognition was hacked[10]. See `http://www.dailytech.com/Hackers+Make+Short+Work+of+SuperSecure+Facial+Biometrics/article14316.htm`, retrieved November 2016

[2]`http://www.tabularasa-euproject.org/`, retrieved November 2016

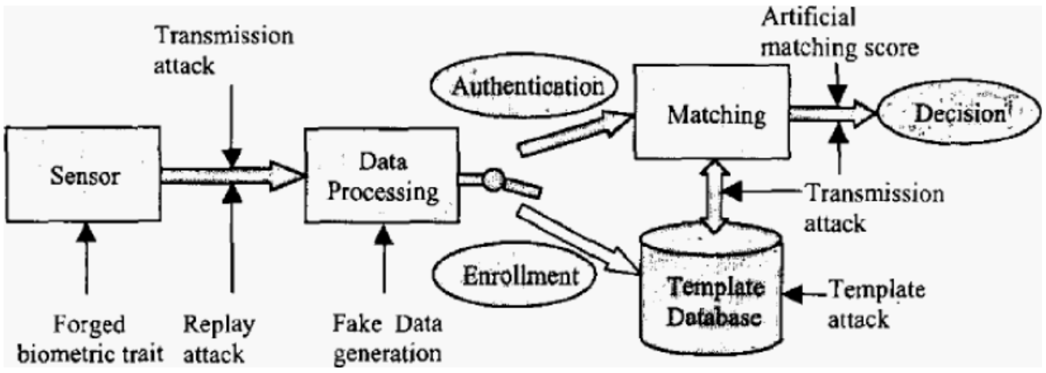


Figure 1. The different attacks on biometric authentication systems[33].

that the device itself moves (and therefore there will be more background motion than on fixed devices).

To increase the spoofing detection capabilities on mobile devices, this research extends an existing spoofing detection method that does not rely on certain types of hardware or camera capabilities. The existing methods is presented by De Marsico et al. [7] and detects spoofing by checking if the presented face is actually a 3D face or just a 2D representation of a face. Unfortunately, this method can only detect spoofing attacks that use photos. Furthermore this method relies on the user rotating its face, while this is not enforced in any way.

To overcome these problems, we extend this method by adding a challenge for the user, for which it has to rotate its head. This ensures that there is rotation of the head, which is needed for the check of 3D properties. Furthermore, this challenge makes spoofing attacks that use videos harder, because the head has to rotate in a specific pattern.

Tests on this challenge show that users are capable of completing the challenge successfully, and that it can detect different types of spoofing attacks with high accuracy rates.

2. RELATED WORK

Face spoofing detection has been researched since at least 2004 [19]. Since then much research has been done, of which an overview is given by e.g. Galbally et al. [9]. Most spoofing detection methods can be categorized in two groups:

- liveness / motion detection
- texture / image quality detection

Some methods do not fall within these groups, like ones which check for the edges of a spoofing medium or those which check for 3D-properties of the face.

2.1 Liveness / Motion Detection

Liveness detection distinguishes between real faces and photos by checking if the presented face is static or not. Well known examples are spoofing detection methods which check for eye-blinking [11] or eye-moving [25]. This is a robust and intuitive way of detecting spoofing attacks that use photos, but can easily be circumvented by e.g. cutting eye-holes in images.

Other methods check for liveness in the entire face, for example by using Optical Flow Fields [3][14]. These methods check for movement within the face caused by change of facial expressions or for the difference in movement between specific regions of the face, caused by some regions being closer to the camera than others.

Another type of spoofing detection using motion is checking for difference of movement within the face, compared to the close surroundings of the face [34]. If there is much similarity in these regions, than probably an image is shown to the camera which contains both the face as well as the detected background.

All these methods are able to detect spoofing attacks with photos to some extend, but there are some issues:

- they do not work against spoofing attacks which uses videos;
- small changes to the photos – like cutting eye-holes or bending/folding the photo – do decrease the performance of many of these spoofing detection methods.

2.2 Texture / Image Quality Detection

Methods in this category try to measure differences in image-details between real faces and the spoofing attack surfaces, like computer screens or paper. This is based on multiple assumptions, like that recapturing footage leads to decrease of quality; that faces reflect light in a different way than other surfaces or that printing on paper creates detectable artefacts.

One of the most popular methods of this type is based on the use of Local Binary Patterns (LBP) [21]. This method analyses micro-textures to distinguish between spoof and non-spoof videos. This allows for the detection of print-artefacts, quality degradation of footage and differences in reflection. In this way, photos, videos or masks can be detected [6, 8]. Because LBP scores good results on the detection of spoofing attacks, many variations are presented. Some of these use different texture-descriptors, like LPQ [1], WLD [23] or TLP [26]. Other methods try to add frame-transcending information to LBP by using e.g. LBPV [18] or LBP-TOP [15]. Most of these method improve the performance of LBP by some extend. The popularity of LBP results in tests on the spoofing detection capabilities of LBP varying from spoofing attacks with pictures to spoofing attacks using 3D masks.

However, recent research using cross-database testing shows that the results of LBP decrease significantly when the training-footage is from another database then the test-footage [20]. This indicates that the performance of LBP on mobile phones may be pretty poor, because just like cross-database tests, biometric authentication on mobile phones cause great differences per situation. Furthermore, most mobile phones contain very basic cameras, which may influence the detected micro-textures and therefore the results of LBP.

Other methods that can be categorized in the group of texture / image quality detection methods are methods using high frequency components [2, 19, 30]. These methods use the reflection properties of the captured surface to distinguish between genuine authentication attempts and spoofing attempts. These methods are sensitive to differing lighting conditions, so filters are applied to decrease the effects of lighting [27]. Still, the results of these kind of methods remain behind those of LBP.

2.3 Other Methods

Next to the methods using texture information or liveness detection, other approaches are proposed. For example Komulainen et al.[16] presented a method which is focused on detecting the spoofing medium. A combination of two techniques is used to achieve this:

- an upper-body detector is used to see if the upper-body is in line with the face;
- histograms of gradients are used to see if there are distinct edges in the footage, probably caused by the edges of a screen or photo.

This method is a good addition to existing methods, but because it only works in specific conditions, it is easy to spoof when it works on its own.

Another approach is presented by Smith et al.[29]. For this method the screen of a phone emits a certain colour for a short time. Then it is checked if this colour was reflected by the face, to see if there was some replay attack going on. Unfortunately they had to conclude that most phones do not have a screen that is bright enough to perform this method.

Fusion of methods.

Because one method is not yet sufficient to counter spoofing attacks, many researchers try to fuse multiple methods in order to get better results. A set of different combinations that are tried:

1. Motion and texture[4][17]
2. Frequency and texture[13]
3. Different texture methods[22]

Combining different methods does in many cases increase the performance compared to using only a single detection method.

Spoofing detection for mobile phones can still be improved. Many methods only detect certain types of spoofing, can easily be tricked or require some specific constraints like fixed cameras. With the method presented in this paper, we aim to extend the range of existing methods with a method suitable for using on mobile phones. For this we extend an existing method to overcome some of its limitations.

3. 3D-DETECTION

Another approach of detecting spoofing that does not directly rely on texture information or liveness detection is proposed by De Marsico et al. [7] and Wang et al. [31]. Their methods slightly differ, but both methods detect spoofing attacks by assessing the 3D properties of a face. The methods rely on the assumption that when a real face rotates, the landmarks on the face move in a different way then when a printed face rotates (see Figure 2). By taking a set of landmarks and calculating the ratio of the distance between them, 2D representations of faces can be distinguished from 3D faces. Although 3D-detection can successfully be used to detect spoofing using photos of faces, it cannot detect spoofing using videos or 3D masks. This is due to the fact that both of these methods show rotation in a 3D way. The possibility to detect bended photos or photos with the nose cut out is not yet evaluated very well.

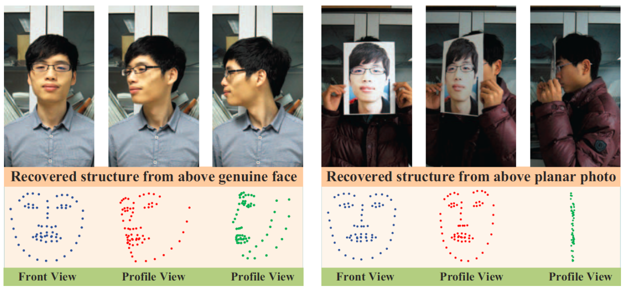

Recovered structure from above genuine face
Recovered structure from above planar photo
Front View
Profile View
Profile View
Front View
Profile View
Profile View

Figure 2. A comparison of how landmarks behave between a genuine face and the photo of a face [31].

3.1 Landmark Detection

In order to perform 3D detection, face-landmarks have to be extracted from the given footage. The distances between these landmarks will be used later to distinguish between genuine faces and spoofing attempts. Two different landmark detection algorithms have been tested and compared.

3.1.1 STASM

STASM [24] is a landmark detection algorithm that is able to detect 77 different landmarks on the face by using the following steps:

1. The location of the face is determined using a global face detector.
2. The rough location of the landmarks is determined using the output of step 1.
3. Iterate through the following steps about four times:
   (a) For each landmark:
      i. Get the portion of the image surrounding that landmark;
      ii. Compare this patch to known sample patches for this landmark;
      iii. Improve the positioning of the landmark.
   (b) Confirm the newly suggested shape of all landmarks, by comparing it to a global shape model.

On the setup we used, STASM was able to process about 8 images per second. Remarkable is that when STASM is presented a picture multiple times, the position of the landmarks slightly differed. This caused inaccuracy in the detection of the landmarks by STASM.

3.1.2 DEST

DEST[3] works somewhat similar as STASM. However, instead of improving the position of landmarks based on a

[3]Deformable Shape Tracking: `https://github.com/cheind/dest`. Retrieved November 2016

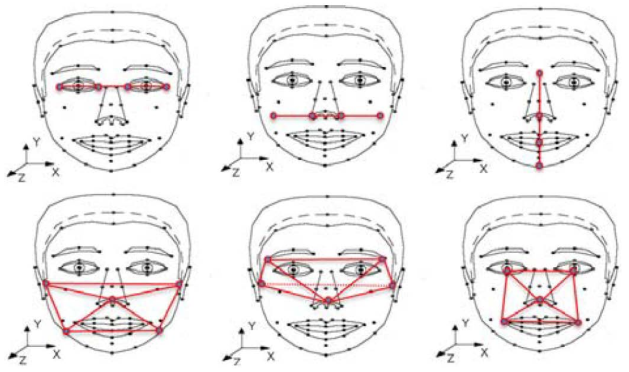


Figure 3. The configuration of landmarks used by De Marsico et al. [7]. The three upper configurations are collinear ($c_1 - c_3$), the three lower configurations are coplanar($c_4 - c_6$).

global shape model, it uses regression trees to do so [12]. This increases the performance for rotated faces compared to STASM. DEST locates 64 landmarks and on our system it was able to process about 3 images per second.

3.1.3 Comparing STASM and DEST

As shown by Kazemi et al. [12], DEST offers better performance on recognizing faces. This is caused by i.e. better recognition of landmarks for rotated faces. STASM however, provides a better frame rate than DEST. STASM is used in the original 3D-detection method of De Marsico et al. [7]. For their research the accuracy of STASM was sufficient. However, some limitations of STASM made it unsuitable for our research:

1. The inaccuracy in the localizations of landmarks resulted in a poor user experience when trying to perform the challenge.

2. We want the user to rotate their head, but because STASM uses predefined models for alligning the landmarks, it does not handle rotated faces very well, resulting in:

    - Worse accuracy of landmark positioning for rotated faces.
    - A small angle of rotation for which faces are recognized.

Because DEST does not have these limitations, DEST is used for the rest of our research.

3.2 Calculating 3D Movement

The method of De Marsico et al. [7] is used to recognize 2D spoofing. This method uses the cross-ratios of different configurations of landmarks to differentiate between 3D and 2D movement (see Figure 3). In theory, the cross-ratio of the collinear points should stay the same for any rotation of both 2D and 3D faces. However, the cross-ratio of the coplanar points should change when a 3D face is rotated, while it should stay the same for a 2D face. Their research shows that this method can successfully be used to distinguish between real faces and printed faces. This is done by measuring the change of the value for the cross-ratios over time: if the change of these values is above a certain threshold, a face is marked as genuine. The precise calculation of the cross-ratio can be found in [7].

In this research, some changes are made in calculating 3D movement compared to the method presented by De Marsico et al.. First of all, the configuration of landmarks is chosen slightly different, because some of the points shown in Figure 3 do not correspond to landmarks used by STASM or DEST. This means that we picked the landmarks that were close to what can be seen in Figure 2. Second, we did not use the configurations c_1, c_2 and c_3. In the original research, these points were just used as a test to prove the initial assumption that these points would not change when rotating the face. Therefore they are not relevant to this research.

Thirdly, De Marsico et al. present two different formulas to calculate cross-ratios for the configurations c_4 to c_6. How they exactly combine these formulas is not explained in their paper. Therefore, we chose to use both cross-ratios separately, which gives us 6 different cross-ratios to work with.

After determining the cross-ratios for each frame, their variation over time should be measured, since this shows if the cross-ratios change over time. This is done in a similar way as shown by De Marsico et al, like shown in Algorithm 1. There are 5 variables in this algorithm that have to be set:

k is the range over which mean and variation are calculated. This variable is also used by De Marsico et al.. They show that larger values of k result in better results. However, picking k large does also cause more complex calculations and the need for longer videos. Because De Marsico et al. got good results for $k = 25$, this value is chosen for this research too.

t_{start} is set to the begin of the challenge $+ k$ frames. Adding these frames is done because the first frames cannot be compared to k frames before them, since they are not part of the challenge.

t_{end} is the end of the challenge.

th_j is the set of thresholds that determine if the variation of a cross-ratio is probably caused by 3D movement. For each cross-ratio, another threshold is set. For this research, the thresholds are trained using some self-made footage.

th_v is the threshold which determines how many of the frames should show '3D-movement'. This threshold is trained with the same videos as used for training th_j. It is set to 80%.

4. CHALLENGE

The given method for spoofing detection relies on the user rotating its head. In order to force the user to do this, we add a challenge to the existing system. This challenge is inspired by the swipe patterns that are often used on android smartphones. It requires the user to move a dot on the screen in a given pattern by rotating its head. If the user succeeds in doing this, we consider the challenge succeeded, but if the user deviates from the pattern or runs out of time, the challenge is failed. An additional advantage of adding this type of challenge to the current system is that asking for a specific random generated pattern makes the challenge more robust against other types of attack such as pre-recorded videos. This is because in these videos, the user does not rotate its head in the specific directions and order which the challenge asks for.

Algorithm 1 Algorithm for calculating 3D movement.

```
1: procedure CALCULATE 3D MOVEMENT
2:     for frame $I$ do
3:         extract landmarks $L$ from $I$
4:         compute cross ratio $c_j$ from $L$
5:         for $z \leftarrow$ frame $t_{start}$ to $t_{end}$ do    ▷ $t_{start}$ and $t_{end}$ are respectively the begin and end of the challenge
6:             $m_j \leftarrow (1/k) \sum c_j$    ▷ mean; $k$ is set to 25
7:             $v_j \leftarrow (1/k) \sum |c_j - m_j|$    ▷ variation
8:         if $v_j > th_j$ then    ▷ $th_j$ differs for each cross-ratio
9:             $genuine = genuine + 1$
10:    if $genuine/y - x > th_v$ then    ▷ $th_v$ is set to 80%
11:        return "Genuine"
12:    else
13:        return "Spoof"
```

4.1 Calculating Rotation

The first step required in the design of the challenge is measuring the rotation of the head of the user. This will allow the user to move the point that will be presented to him during the challenge. It is important to note that we will only use yaw (shaking) and pitch (nodding) rotations for the challenge. This is because using roll rotation is less user-friendly (it is not a movement people used very often), but mostly because roll rotation does not change the ratios of distances between landmarks. So roll rotation would not help when trying to measure spoofing attacks based on 3D-movement.

The principles used for measuring '3D-movement will also be used for measuring rotation. Namely that the ratios of the distances between landmarks vary when a head is rotated. Two different methods are implemented to measure rotation and both use the landmarks that are detected by DEST.

4.1.1 Using the Cross-Ratios

The first method uses the cross-ratios that are also used to calculate '3D movement'. Since rotating the head influences this cross-ratios, they can be used to determine the current rotation of the head. To do this, first an image is taken on which the users head is facing the camera. The cross-ratios in this image act as a reference to all other frames that will be processed. When the user then rotates his face, the cross-ratios are changing too, but they all react differently to different kinds of rotation. For example the first cross-ratio for c_5 reacts about ten times stronger on yaw rotation than on pitch rotation. By combining different combinations for yaw and pitch rotation, we are able to determine the rotation of a given face quite accurate. The used formulas are:

$$pitch = \frac{c_{4cp_1} + c_{5cp_1} - c_{5cp_2}}{3} \quad (1)$$

$$yaw = \frac{c_{5cp_1} - c_{4cp_2} + c_{5cp_2} + c_{6cp_2}}{4} \quad (2)$$

Where for c_x corresponds to the different configurations of landmarks from Figure 3 and cp_y corresponds to the different cross-variants described in [7].

4.1.2 Using Nose-Eye-Distance

The second method for calculating the rotation of the face is somewhat more straightforward. This method uses the distance between the nose and both eyes to determine the rotation of the face. Yaw rotation can be measured, because, for the view of the camera, the nose seems to move faster than the eyes. Therefore, when the face is turned to the left, the nose is closer to the left eye and vice versa. Pitch rotation can be measured because when the face is turned up, the tip of the nose moves between the eyes. When the face is turned down, at least for the first degrees of rotation the distance between the nose and the eyes grows for the viewpoint of the camera. Because we do not need extreme downward movement from a user, this increase in distance for the first degrees of rotation is sufficient. Like for the other method, an image is taken where the user is looking straight to the camera as a reference point. The used formulas are:

$$d = \frac{x_{nose} - x_{lefteye} + x_{nose} - x_{righteye}}{2} \quad (3)$$

$$pitch = \frac{d_{currentframe} - d_{reference}}{d_{reference}} \quad (4)$$

and:

$$d = |y_{nose} - y_{lefteye}| - |y_{nose} - y_{lefteye}| \quad (5)$$

$$yaw = d_{current_frame} - d_{reference} \quad (6)$$

This method also results in accurate rotation tracking. However, because using cross-ratios does sometimes generate inaccurate results when the nose is in line with other landmarks, we chose to use the nose-eye-distance to calculate the rotation for the challenge.

4.2 Challenge design

Using the ability to calculate the rotation of the face of a user, a challenge can be presented. The challenge presents the user with a 3x3 grid, which is familiar for many people because of the swipe pattern that can be used to unlock android phones. Next to this grid, the user is shown a red dot, which he can move by rotating his face. Yaw rotation causes movement of the dot over the y-axis and pitch rotation over the x-axis. The aim of the challenge is to let the red dot follow a determined path which consists of multiple nodes on the grid. This path is randomly generated, and consists of 4 nodes. The node to which the red dot has to be moved is highlighted in the grid. When the red dot gets within a set range around the node that it has to move to, it can proceed to the next node. However, if the red dot gets within the range of a node that it should not move to, the challenge is stopped and the user is informed that the challenge is failed.

The pattern is generated by selecting one random node on the grid. Hereafter, an adjacent horizontally or vertical neighbour is selected as the next node, and so on, until 4 nodes are chosen. Diagonal neighbours are not used, because this makes it hard for the user to move from one node to another without getting into the range of a third node.

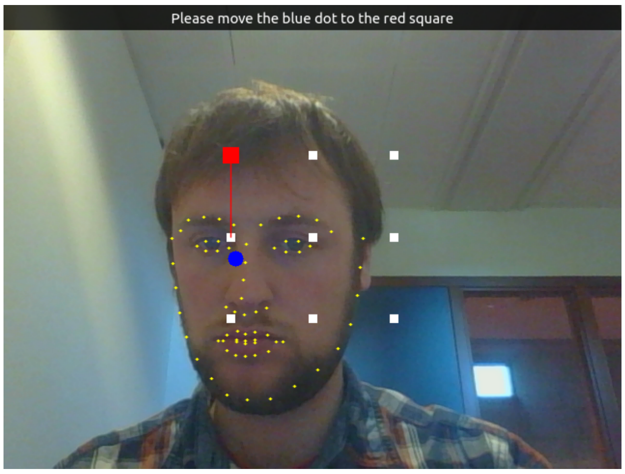


Figure 4. An example of how the challenge is presented to the user.

The range around the nodes to which the red dot has to move can be varied. A higher range makes it easier for the user to reach a dot, but at the same time, inaccurate steering of the red dot will result into getting near a wrong dot faster. The rotation needed to get the red dot near the edges of the grid can also be fluctuated. When more rotation is needed, more 3D-movement of the face can be measured and inconsistency of the landmark detection gets less influential. However, letting the user rotate its head far is bad for the user experience and large rotations will also decrease the performance of the landmark detection, since most of them perform best on frontal faces. We chose to set the radius around the nodes about 1/5th of the size of the grid and the movement needed to get to the edge of the grid at about 5 degrees rotation.

Because the challenge uses a random pattern, it makes spoofing using videos harder. By using 4 nodes to create a pattern, 192 different patterns are possible. If more complex patterns are required, extra nodes can be added, for which each extra node will multiply the complexity by about 3.

4.3 Judging recorded footage

After successfully completing a challenge, the authentication attempt cannot automatically be marked as genuine. Tricks could be used to complete the challenge, like using 9 pictures of a person looking in different directions corresponding to the rotation needed to move the dot to a node of the challenge. To overcome these kind of attacks, the challenge is recorded and this footage is used to check for inconsistencies. This is done by assessing if the landmark detection is able to reliably detect the face in a majority of the frames, by checking:

1. If the landmark detection detects a face at all.
2. If the location of the landmarks does not change rigidly between frames.

This second check is performed because rigid movement of landmarks typically indicate that either the landmark detection does detect something other than a face – e.g. because there is no face or because a face is rotated too much – or because there are multiple faces visible in the frame. However, if a picture is shown to the camera on a screen and this picture is suddenly changed, then rigid movement in the location of landmarks would occur too.

A frame is labeled containing too much movement of the landmarks when at least x landmarks move more then a certain threshold compared to the last reliable frame. If multiple frames in a row are marked, the thresholds are slowly raised. These thresholds are individually set per landmark and are trained by using the training set of our dataset. When more than 20 percent of the frames in the footage contain questionable landmarks or no detected landmarks at all, the video is labeled as imposter.

4.4 Overview

An overview of the complete system is shown in Figure 5. The user is first presented with the challenge described in section 4.2 (steps 1 & 2). This step will generate movement of the face, required for step 5 and is expected to filter out all spoofing attacks using still images and unmodified videos. If this challenge is performed successfully, the recorded footage is evaluated based on the accuracy of the landmark detection as described in section 4.3 (steps 3 & 4). This step will filter out videos which cannot be reliably processed. We expect that this step will also filter out some types of attacks – like ones which uses pre-selected photos of rotated pictures. The videos that have passed the challenge and the accuracy check are then tested on 3D-movement, like described in section 3.2 (step 3 & 5). Videos passing all these tests are labeled as genuine authentication attempts, the rest is labeled imposter.

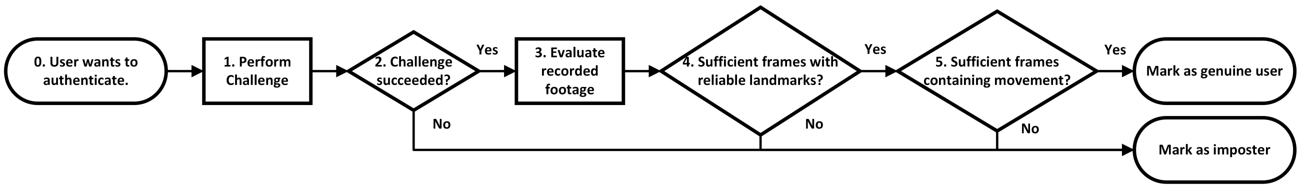
5. FUSION WITH TEXTURE ANALYSIS

A typical method to increase the performance of a spoofing detection algorithm is to fuse multiple algorithms together. Since our algorithm is probably not capable of detecting spoofing attacks using masks or real-life manipulated videos it would make sense to fuse it with a method that will likely detect these kinds of attacks. Only few methods are tested on masks, but one method proved to be able to detect masks and videos is LBP-TOP [6]. We tried to combine LBP-TOP with 3D-movement detection, but ran into some issues of which some can be fixed and some pose a greater problem. First of all, the code that was intended to be used, based on the research done by the IDIAP institute[5], was not properly maintained[4]. This could be overcome, just like the fact that its training and running procedure was much longer than expected (training on a database could easily cost one or more days). A more permanent issue is the difficulty of using LBP-TOP on footage captured on mobile devices. Recent research shows that the performance of LBP drops significantly when tested using inter-DB testing compared to tests which train and test on the same database [20, 32]. Especially for usage on mobile phones – where very many different types of cameras are used and the environment cannot be controlled easily – the micro-textures which are used by LBP-TOP may vary heavily, which will decrease the performance of the algorithm. This, in combination with the badly maintained software-library, caused us not to implement a fusion with LBP-TOP.

6. EVALUATION

The combination of a challenge and calculating 3D movement is evaluated to test if this is a usable method for spoofing detection. The aim of the evaluation is to do a proof-of-concept. The algorithm is tested on two properties: usability and spoofing detection rate.

[4]`https://pypi.python.org/pypi/antispoofing.lbptop/2.0.3`, retrieved November 2016



0. User wants to authenticate.
1. Perform Challenge
2. Challenge succeeded?
Yes
No
3. Evaluate recorded footage
4. Sufficient frames with reliable landmarks?
Yes
No
5. Sufficient frames containing movement?
Yes
No
Mark as genuine user
Mark as imposter

Figure 5. An overview of the complete spoofing detection method.

To test usability, real users are presented with the challenge multiple times. It is then measured how many attempts to perform the challenge are successful. In order for the challenge to be usable, each user should be able to finish the challenge successfully multiple times. This will show that when the user is faced with the challenge in real-life, he will be able to finish it successfully at most after trying a few times.

To test the spoofing detection rate, the algorithm is tested on both footage of both genuine and imposter authentication attempts. To get footage for this test, a new dataset is created (see 6.1). The detection rate is measured by comparing the scores for genuine and imposter authentication attempts and by comparing the influence of different steps in the process with each other. For the performance of the challenge, all of the footage of the dataset can be used, since the challenge was not trained on this data. However, for the performance of checks on the reliability of the landmarks and the 3D-movement, only the test-data of the dataset is used.

6.1 The used dataset

For testing the performance of our proposed challenge, footage is needed where users try to perform this challenge. Unfortunately, as far as we know, no database currently meets the need of people rotating their head for certain degrees and certain times. Therefore, we have to create our own dataset. This dataset exists of 90 videos of genuine authentication attempts and 30 imposter ones.

The videos for the genuine authentication attempts are created using 9 different subject. Each subject first got 1 minute to get accustomed to the movement of the dot on the screen by rotating their head. Hereafter, they were asked to try to perform the challenge 10 times. The environment in which the videos were made was partially controlled, meaning that the lighting was quite even on the face and there were no exceptionally light or dark environments. The backgrounds do differ for different subjects ranging from white walls to ordinary living rooms. The footage was captured using the default webcam of a Lenovo Thinkpad W540.

The spoofing videos are created using 2 different photos. One photo is captured from a frame of one of the genuine authentication attempts and the other photo was shot using a Canon EOS 550D. Three different scenarios are created using these images:

1. Holding the flat printed photo in front of the camera and trying to finish the challenge by rotating the photo.

2. The same as point 1, but than the photo is freely folded and bend to try to create the desired 3D effect.

3. The same as point 2, but the edged of the nose are cut and bend, so the nose is closer to the camera than the rest of the photo.

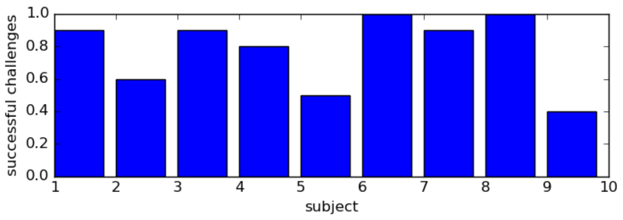


Figure 6. Percentage of successful challenges per subject.

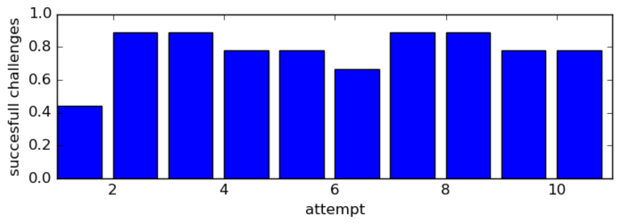


Figure 7. Percentage of successful challenges per attempt.

For each scenario 10 attempts are made and captured on video. The environments in which the videos are made are the same as for the videos of the genuine authentication attempts.

The whole dataset is randomly divided into a training-set and a test-set. The training-set consists of 67% of the footage, the rest is in the test-set. For testing the challenge only, both sets could be used (since the challenge was not trained on the training-set of the dataset).

6.2 Results

The results for testing the usability of the spoofing detection method are shown in Figure 6, 7, 8 and 9. These graphs show that on average 78% of the challenges are completed successfully. Interesting is that the first challenge failed for 56% of the subjects, while the other challenges failed on average for only 19% of the subjects. Subject 9 failed the most of the challenges, and as can be seen in Figure 8 and Figure 9 this person tried to perform the challenge exceptionally fast, with both fast times for successful and failed challenges. Observing the subjects

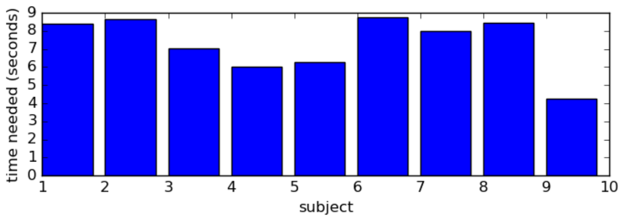


Figure 8. Time needed for successful challenge.

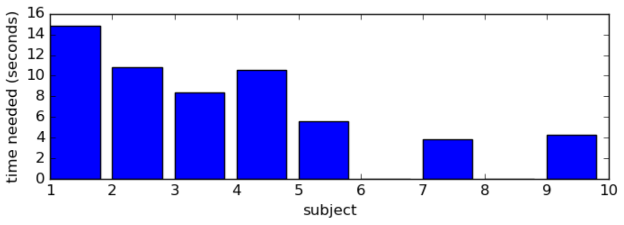


Figure 9. Time needed for failed challenge.

performing the challenge, it may very well be that people who were more patience did also achieve better results. For the failed challenges, 2 failed because time ran out and the other 18 challenges failed because the dot on the screen was moved to a wrong node.

The results for the spoofing detection are shown in Table 1. Since none of the imposters are labeled as a genuine user, the FAR is 0%. The FRR is 52%, resulting in a HTER of 24%. The score for 3D movement is very interesting, since it only increases the FRR, while all imposter videos are already discarded before this test is performed. If this test would be removed from the setup, the FRR would decrease to 35%, which would result in an HTER of 18%.

The test shows that the majority of the imposters fail to complete the challenge. For the test-set this is 78%. However, since the challenge is not trained on the training-set of our database, we are also able to run the challenge on both the training-set and the test-set. When we do this, the FRR of the challenge drops from 28% to 22% and the rejected imposters increase from 67% to 87%. This major difference is caused because almost all imposter videos with a successfully completed challenge are in the test-set.

7. CONCLUSION & DISCUSSION

As stated earlier, the evaluation that is performed on our system is done to make a proof of concept. The results from the evaluation provide a good indication of what this method is capable of, but cannot be used for a detailed comparison of FAR's and FRR's. With this in mind, the results show that the concept of a challenge is certainly a good addition to a spoofing detection algorithm and might even work well on its own. With 0% FAR on the current configuration and test-set, it is able to detect spoofing attempts using photos very well. At the same time, it is viable for users to perform the challenge successfully, certainly when multiple tries are given.

Since the results of this research are promising, it is advised to test the challenge on a more diverse range of attacks. For example videos that are manipulated specifically to pass the challenge and 3D-masks can provide interesting testing material. Also, the development of the challenge would benefit from development and training with a larger and more diverse dataset. For example, for this research no people with a 'flat' face structure (meaning the nose is not much nearer to the camera than the eyes) were in the dataset.

The robustness of this method is a major advantage compared to other spoofing detection methods. Influences of lighting, environment and camera quality are limited, because the method only relies on accurate landmark detection. However, a slightly larger amount of user-collaboration is needed, since the user needs to rotate his face to complete the challenge. For use-cases where user-collaboration is not a problem, the presented challenge can provide a biometric authentication system with a decent level of security.

8. REFERENCES

[1] S. R. Arashloo, J. Kittler, and W. Christmas. Face spoofing detection based on multiple descriptor fusion using multiscale dynamic binarized statistical image features. *Information Forensics and Security, IEEE Transactions on*, 10(11):2396–2407, 2015.

[2] J. Bai, T.-T. Ng, X. Gao, and Y.-Q. Shi. Is physics-based liveness detection truly possible with a single image? In *Circuits and systems (ISCAS), Proceedings of 2010 IEEE international symposium on*, pages 3425–3428. IEEE, 2010.

[3] W. Bao, H. Li, N. Li, and W. Jiang. A liveness detection method for face recognition based on optical flow field. In *Image Analysis and Signal Processing, 2009. IASP 2009. International Conference on*, pages 233–236. IEEE, 2009.

[4] I. Chingovska, A. Anjos, and S. Marcel. Anti-spoofing in action: joint operation with a verification system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 98–104, 2013.

[5] T. De Freitas Pereira, A. Anjos, J. M. De Martino, and S. Marcel. Lbp-top based countermeasure against face spoofing attacks. In *Computer Vision-ACCV 2012 Workshops*, pages 121–132. Springer, 2012.

[6] T. De Freitas Pereira, J. Komulainen, A. Anjos, J. M. De Martino, A. Hadid, M. Pietikäinen, and S. Marcel. Face liveness detection using dynamic texture. *EURASIP Journal on Image and Video Processing*, 2014(1):1–15, 2014.

[7] M. De Marsico, M. Nappi, D. Riccio, and J.-L. Dugelay. Moving face spoofing detection via 3d projective invariants. In *Biometrics (ICB), 2012 5th IAPR International Conference on*, pages 73–78. IEEE, 2012.

[8] N. Erdogmus and S. Marcel. Spoofing face recognition with 3d masks. *Information Forensics and Security, IEEE Transactions on*, 9(7):1084–1097, 2014.

[9] J. Galbally, S. Marcel, and J. Fierrez. Biometric antispoofing methods: A survey in face recognition. *Access, IEEE*, 2:1530–1552, 2014.

[10] A. Hadid. Face biometrics under spoofing attacks: Vulnerabilities, countermeasures, open issues, and research directions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 113–118, 2014.

[11] H.-K. Jee, S.-U. Jung, and J.-H. Yoo. Liveness detection for embedded face recognition system. *International Journal of Biological and Medical Sciences*, 1(4):235–238, 2006.

[12] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014.

[13] G. Kim, S. Eum, J. K. Suhr, D. I. Kim, K. R. Park, and J. Kim. Face liveness detection based on texture and frequency analyses. In *Biometrics (ICB), 2012 5th IAPR International Conference on*, pages 67–72. IEEE, 2012.

[14] K. Kollreider, H. Fronthaler, and J. Bigun. Non-intrusive liveness detection by face images. *Image and Vision Computing*, 27(3):233–244, 2009.

Table 1. Spoofing detection rates on test-set. The tests are sequential. So e.g. if a video fails the challenge, it will not be scored on the accuracy of landmarks and 3D-movement.

	Labeled genuine	Failed challenge	Inaccurate landmarks	Insufficient 3D movement
Genuine	**48%**	**28%**	**7%**	**17%**
Imposter - flat images	0%	100%	0%	0%
Imposter - bend images	0%	67%	33%	0%
Imposter - cut-out nose	0%	67%	33%	0%
Imposter - total	**0%**	**78%**	**22%**	**0%**

[15] J. Komulainen, A. Hadid, and M. Pietikäinen. Face spoofing detection using dynamic texture. In *Computer Vision-ACCV 2012 Workshops*, pages 146–157. Springer, 2012.

[16] J. Komulainen, A. Hadid, and M. Pietikainen. Context based face anti-spoofing. In *Biometrics: Theory, Applications and Systems (BTAS), 2013 IEEE Sixth International Conference on*, pages 1–8. IEEE, 2013.

[17] J. Komulainen, A. Hadid, M. Pietikainen, A. Anjos, and S. Marcel. Complementary countermeasures for detecting scenic face spoofing attacks. In *Biometrics (ICB), 2013 International Conference on*, pages 1–7. IEEE, 2013.

[18] N. Kose and J.-L. Dugelay. Classification of captured and recaptured images to detect photograph spoofing. In *Informatics, Electronics & Vision (ICIEV), 2012 International Conference on*, pages 1027–1032. IEEE, 2012.

[19] J. Li, Y. Wang, T. Tan, and A. K. Jain. Live face detection based on the analysis of fourier spectra. In *Defense and Security*, pages 296–303. International Society for Optics and Photonics, 2004.

[20] S. Liu, B. Yang, P. C. Yuen, and G. Zhao. A 3d mask face anti-spoofing database with real world variations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 100–106, 2016.

[21] J. Määttä, A. Hadid, and M. Pietikainen. Face spoofing detection from single images using micro-texture analysis. In *Biometrics (IJCB), 2011 international joint conference on*, pages 1–7. IEEE, 2011.

[22] J. Maatta, A. Hadid, and M. Pietikainen. Face spoofing detection from single images using texture and local shape analysis. *Biometrics, IET*, 1(1):3–10, 2012.

[23] L. Mei, D. Yang, Z. Feng, and J. Lai. Wld-top based algorithm against face spoofing attacks. In *Biometric Recognition*, pages 135–142. Springer, 2015.

[24] S. Milborrow and F. Nicolls. Active shape models with sift descriptors and mars. In *VISAPP (2)*, pages 380–387, 2014.

[25] G. Pan, L. Sun, and Z. Wu. *Liveness detection for face recognition*. INTECH Open Access Publisher, 2008.

[26] S. Parveen, S. M. S. Ahmad, N. H. Abbas, W. A. W. Adnan, M. Hanafi, and N. Naeem. Face liveness detection using dynamic local ternary pattern (dltp). *Computers*, 5(2):10, 2016.

[27] B. Peixoto, C. Michelassi, and A. Rocha. Face liveness detection under bad illumination conditions. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 3557–3560. IEEE, 2011.

[28] D. F. Smith, A. Wiliem, and B. C. Lovell. Binary watermarks: a practical method to address face recognition replay attacks on consumer mobile devices. In *Identity, Security and Behavior Analysis (ISBA), 2015 IEEE International Conference on*, pages 1–6. IEEE, 2015.

[29] D. F. Smith, A. Wiliem, and B. C. Lovell. Face recognition on consumer devices: Reflections on replay attacks. *Information Forensics and Security, IEEE Transactions on*, 10(4):736–745, 2015.

[30] X. Tan, Y. Li, J. Liu, and L. Jiang. Face liveness detection from a single image with sparse low rank bilinear discriminative model. In *Computer Vision–ECCV 2010*, pages 504–517. Springer, 2010.

[31] T. Wang, J. Yang, Z. Lei, S. Liao, and S. Z. Li. Face liveness detection using 3d structure recovered from a single camera. In *Biometrics (ICB), 2013 International Conference on*, pages 1–6. IEEE, 2013.

[32] D. Wen, H. Han, and A. K. Jain. Face spoof detection with image distortion analysis. *Information Forensics and Security, IEEE Transactions on*, 10(4):746–761, 2015.

[33] Q. Xiao. Security issues in biometric authentication. In *Information Assurance Workshop, 2005. IAW'05. Proceedings from the Sixth Annual IEEE SMC*, pages 8–13. IEEE, 2005.

[34] J. Yan, Z. Zhang, Z. Lei, D. Yi, and S. Z. Li. Face liveness detection by exploring multiple scenic clues. In *Control Automation Robotics & Vision (ICARCV), 2012 12th International Conference on*, pages 188–193. IEEE, 2012.