Secure Face Unlock: Spoof Detection on Smartphones

Keyurkumar Patel, Student Member, IEEE, Hu Han, Member, IEEE, and Anil K. Jain, Fellow, IEEE

Abstract—With the widespread deployment of face recognition systems in applications from de-duplication to mobile device unlocking, security against face spoofing attacks requires increased attention; such attacks can be easily launched via printed photos, video replays and 3D masks of a face. We address the problem of face spoof detection against print (photo) and replay (photo or video) attacks based on the analysis of image distortion (e.g., surface reflection, moiré pattern, color distortion, and shape deformation) in spoof face images (or video frames). The application domain of interest is smartphone unlock, given that growing number of smartphones have face unlock and mobile payment capabilities. We build an unconstrained smartphone spoof attack database (MSU USSA) containing more than 1,000 subjects. Both print and replay attacks are captured using the front and rear cameras of a Nexus 5 smartphone. We analyze the image distortion of print and replay attacks using different (i) intensity channels (R, G, B and grayscale), (ii) image regions (entire image, detected face, and facial component between the nose and chin), and (iii) feature descriptors. We develop an efficient face spoof detection system on an Android smartphone. Experimental results on the public-domain Idiap Replay-Attack, CASIA FASD, and MSU-MFSD databases, and the MSU USSA database show that the proposed approach is effective in face spoof detection for both cross-database and intra-database testing scenarios. User studies of our Android face spoof detection system involving 20 participants show that the proposed approach works very well in real application scenarios.

Index Terms—Face antispoofing, face unlock, spoof detection on smartphone, unconstrained smartphone spoof attack database, image distortion analysis

I. INTRODUCTION

With the widespread use of smartphones, biometric authentication, such as face and fingerprint recognition, is becoming increasingly popular for confirming user identity. Two of the most popular smartphone operating systems, Android and iOS, currently use face and fingerprint to authenticate users. With the release of Android 4.0 (Ice Cream Sandwich), Android allows users to unlock their smartphone via facial recognition (FR) technology, and iOS on all iPhones released after the iPhone 5c allows users to unlock their smartphone with their fingerprint (Touch ID). As the use of biometrics for smartphone unlocking and user authentication continues to increase [2], capabilities to detect spoof biometric attacks are needed.

K. Patel and A. K. Jain are with the Department of Computer Science and Engineering, Michigan State University, East Lansing, MI 48824, USA. E-mail: {patelke6, jain}@msu.edu.
H. Han is with Key Lab of Intelligent Information Processing, Chinese Academy of Sciences (CAS), Institute of Computing Technology (ICT), CAS, Beijing 100190, China. E-mail: hanhu@ict.ac.cn.
An earlier version of this paper appeared in the Proceedings of the 8th IAPR International Conference on Biometrics (ICB), Phuket, 2015 [1].
TABLE I

A summary of published methods on 2D face spoof detection.

<table>
<thead>
<tr>
<th>Method</th>
<th>Strength</th>
<th>Limitation</th>
<th>State of the art performance†</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face motion analysis [3]–[6]</td>
<td>Effective for print attack</td>
<td>Requires multiple frames, Slow response</td>
<td>CASIA FASD (Intra-DB, Cross-DB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[20]: (5.82%, 22.6%)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Idiap Replay-attack (Intra-DB, Cross-DB) [7]: (15.54%, 47.1%); [13]: (0.8%, n/a)</td>
</tr>
<tr>
<td>Face texture analysis [8]–[12]</td>
<td>Relatively low computational cost and fast response</td>
<td>Poor generalizability. Requires face and/or landmark detection</td>
<td>Idiap Replay-attack (Intra-DB, Cross-DB) [11]: (5.88%, 35.4%)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CASIA FASD (Intra-DB, Cross-DB) [20]: (12.9%, 43.7%)</td>
</tr>
<tr>
<td>Face 3D shape or depth analysis [9], [16]–[18]</td>
<td>Effective for 2D attacks</td>
<td>Requires multiple frames or additional devices</td>
<td>Idiap Replay-attack [18] (Intra-DB, Cross-DB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[18]: (12.5%, n/a)</td>
</tr>
<tr>
<td>Image quality analysis [19]–[21]</td>
<td>Good generalizability, Low computational cost, Fast response time, Face and/or landmark detection not required</td>
<td>Image quality measures can be device dependent</td>
<td>Idiap Replay-attack (Intra-DB, Cross-DB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[20]: (7.41%, 26.9%); [19]: (15.2%, n/a)</td>
</tr>
<tr>
<td>Frequency domain analysis [10], [22], [23]</td>
<td>Good generalization ability, Low computational cost</td>
<td>Spectral features can be device dependent</td>
<td>CASIA FASD (Intra-DB, Cross-DB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[20]: (14.0%, 38.5%)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>[20]: (5.82%, 22.6%)</td>
</tr>
<tr>
<td>Multi-cue fusion [26], [27] [Proposed]</td>
<td>Good generalizability, Less sensitive to face and/or landmark detection errors, Whole image frame analysis</td>
<td>Moderate computational cost (0.21 sec. on desktop)</td>
<td>Idiap Replay-attack (Intra-DB, Cross-DB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Proposed: (0.0%, 3.5%)* (14.6%, 29.3%)**</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>CASIA FASD (Intra-DB, Cross-DB)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Proposed: (1.67%, 2.5%)* (5.88%, 35.4%)*</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>MSU MFSD [Intra-DB, Cross-DB]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Proposed: (2.67%, 9.27%)* (8.41%, 26.7%)**</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>MSU USSA [Intra-DB, Cross-DB]</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Proposed: (3.84%, 31.4%)</td>
</tr>
</tbody>
</table>

†Intra-database results for the Idiap Replay-Attack, 3DMAD and the UVAD databases are given in terms of Half Total Error Rate (HTER). HTER is defined as the average of false acceptance rate and false rejection rate. Intra-database results for the CASIA FASD, MSU MFSD and the MSU USSA databases are given in terms of Equal Error Rate (EER). For the ZJU Eyeblink and the Private 3D mask datasets, classification accuracy are reported. Cross-database results are given in terms of HTER unless otherwise specified. *Performance was reported using the proposed smartphone protocol; no reject option was used. **Performance was reported using the original intra-database protocol for each database.

a difficult problem that requires continued efforts. A number of the published methods are designed to safeguard the FR system [27] against a specific spoof detection attack, and thus lack good generalizability to different face spoof attacks and application scenarios. Additionally, they are based on databases in which the spoof videos were captured using either low resolution (e.g., webcam) or very high-resolution (e.g., DSLR) cameras [4], [7] (e.g., the CASIA FASD and Idiap databases released in 2012). Therefore, these face spoof databases are not representative of smartphone unlock scenarios. While face spoof detection under the smartphone unlock scenarios was studied in [20], it used a database containing only 50 subjects (videos of only 35 subjects are publicly available). Additionally, results based on a face spoof detection system running on a smartphone platform were not reported.

In this paper, we study the problem of face spoof detection on smartphones using a large unconstrained smartphone spoof attack database, and provide a prototype face spoof detection system running on Android. This paper expands upon our preliminary work [1] in the following ways:

- Collection of a large unconstrained smartphone spoof attack database (MSU USSA) with diverse 2D face spoof attacks (printed photos, and displayed photos) from more than 1,000 subjects to replicate the real scenarios of smartphones unlock.1
- A new feature representation method for face liveness detection by considering the complementarity between different feature cues.
- Study of reject options using IPD constraint and bezel detection to efficiently reject easy cases of spoof attacks.
- Verification of the conclusions drawn in [1] by using the significantly larger MSU USSA database and the inclusion of several new experiments on MSU USSA.
- Promising generalization ability from intra-database to cross-database testing scenarios.2
- Implementation of the proposed method on Android smartphones, and tests in real application scenarios.

1A 10k image portion of the MSU USSA database (where subjects have given approval) will be made available to interested researchers: http://biometrics.cse.msu.edu/pubs/databases.html.
2Cross-database testing involves, training on database A and testing on a different database B, collected in a different setting from database A with different subjects. This is in contrast to the easier, but, not realistic protocol of intra-database testing where, cross-validation is used on a specific database.
The remainder of the paper is organized as follows. In Section II, we briefly review published methods, and 2D face spoof databases. We detail the analysis of image distortions in 2D spoof face images, and the proposed face spoof detection approach on smartphones in Sections III and IV, respectively. Experimental setup, protocols, and results are given in Section V. Finally, we conclude this work in Section VI.

II. RELATED WORK

In this section, we summarize published 2D face spoof detection methods in the literature, give an overview of commonly used 2D face spoof databases, and provide details of the 2D face spoof database that was collected for smartphone unlock scenarios.

A. Literature Review

As summarized in [39], studies on face spoofing detection date back over 15 years. Since then, a number of methods have been proposed for face spoofing detection under print attacks [4], [6], [8], [10], replay attacks [3], [9], [40], and 3D mask attacks [41]. Since our focus is 2D face spoof attack detection (on smartphones), we provide a brief summary and analysis of published 2D face spoof detection methods. Table I groups the published methods into six categories: (i) face motion analysis, (ii) face texture analysis, (iii) face 3D depth analysis, (iv) image quality analysis, (v) frequency domain analysis, and (vi) active methods.

Spoofing detection methods based on face motion analysis extract behavioral characteristics of the face, such as eye blink [4], and lip or head movement [3]. These methods require accurate face and landmark detection to localize the facial components. Additionally, multiple frames must be used in order to estimate the facial motions. These methods are designed to detect print attacks, and thus are not able to handle video replay attacks with facial motions.

Spoofing detection methods based on face texture analysis capture the texture differences (due to different reflection properties of live face and spoof material) between face images captured from live faces and face images captured from various spoof mediums (e.g., paper and digital screen) [7], [8], [43]. These methods can perform spoof detection based on a single face image, and thus have relatively fast response. However, face texture analysis based methods may have poor generalizability when using small training sets with a limited number of subjects and spoofing scenarios.

Spoofing detection methods based on 3D depth analysis estimate the 3D depth of a face to discriminate between 3D live face and 2D spoof face [6], [9]. While live faces are 3D objects, spoof faces presented on 2D planar medium are 2D. Thus, these methods can be quite effective to identify 2D face spoof attacks if the 3D depth information of a face can be reliably estimated. Face 3D depth analysis based methods usually rely on multiple frames to estimate the depth or 3D shape information of a face.

Spoofing detection methods based on image quality analysis utilize the image quality differences between live face images and spoof face images [19], [20], [44]. Since the spoof face images and videos are generated by recapturing live face images and videos in photographs or screens, there will be degradations of color, reflection, and bluriness in the spoof face images compared to the live face images and videos. These methods have been found to have good generalization ability to different scenarios [19]. However, studies on face spoofing detection based on image quality analysis are limited.

Frequency domain based anti-spoofing methods analyze noise signals in recaptured video to distinguish between live and spoof face access [10], [22], [23]. During the recapture of printed photos or video replays, there is a decrease in low frequency components, and an increase of high frequency components. In order to quantize these changes, the input is usually transformed into the frequency domain.

Active methods utilize additional sensors, such as near-infrared (NIR) and 3D depth to capture a face besides the 2D visual face image [24], [25]. While these methods provide better robustness against illumination and pose variations of the face, the use of additional sensors also limit their application scope, particularly in smartphone scenarios.

While many of the published methods belonging to the above five categories report favorable results for intra-database testing, their effectiveness in cross-database testing scenarios, has not been carefully evaluated. The few publications that did conduct cross-database testing tend to report poor results [11], [20], [23]. One plausible approach to improve the robustness of face spoof detection methods under cross-database testing scenarios, is to consider fusion of multiple physiological or behavioral cues [27].

B. 2D Spoof Face Databases

1) Public-domain Databases: In this section, we review the commonly used public-domain 2D face spoof databases in terms of their collection process and their limitations. Additionally, we discuss the database we collected that contains diverse 2D face spoof attacks from a large number of subjects. See Table II.

The Print-attack and Replay-attack databases are both available from Idiap [37]. Live face videos of subjects were captured using the webcam on a MacBook and replay attacks were captured using a Canon PowerShot SX 150 IS camera that records 720p video clips. The high-resolution camera captured replay attacks displayed on an iPhone 3GS (480 × 320 resolution) and iPad I (1024 × 768 resolution).

The CASIA FASD consists of 600 video clips of 50 subjects [40]. Out of the 600 video clips, 150 clips represent video replay attacks. Compared to the Idiap database, the CASIA FASD used a variety of cameras (Sony NEX-5-HD, two low-quality USB) to capture replay attacks displayed on an iPad.

A key limitation of both the Idiap Replay-Attack and CASIA FASD databases is that they captured spoof attacks using either low-resolution cameras (USB Webcam for CASIA FASD), DSLR cameras which are expensive, or leverage spoof mediums that are becoming out-of-date (e.g., iPhone 3GS released in 2009). The low quality cameras used to create the Idiap Replay-Attack and CASIA FASD databases lack autofocus capability, often leading to the capture of unsharp
Face Spoofing Database (MFSD) was collected to study the external camera. In [20], a database named MSU Mobile replay attacks using their built-in cameras instead of an authentication on smartphones.

To replicate the real application scenarios of interest, namely user (them). Hence, using low-resolution or DSLR cameras does not directly above the photo sensor (CCD array in most cameras) as they come equipped with anti-aliasing filters that sit immediately above.

DSLR cameras are different compared to smartphone cameras and low resolution videos. Modern smartphones contain high-resolution front-facing cameras (1.3-megapixels on the Nexus 5 and 8-megapixels on the HTC Desire Eye). Additionally, DSLR cameras are different compared to smartphone cameras as they come equipped with anti-aliasing filters that sit immediately above the photo sensor (CCD array in most cameras) to minimize moiré patterns. These filters reduce the sharpness of an image by smoothing the transitions between pixels, in turn reducing moiré patterns (but not completely eliminating them). Hence, using low-resolution or DSLR cameras does not replicate the real application scenarios of interest, namely user authentication on smartphones.

Smartphones that are equipped with FR systems will capture replay attacks using their built-in cameras instead of an external camera. In [20], a database named MSU Mobile Face Spoofing Database (MFSD) was collected to study the effects of using such videos or images for spoof attacks against smartphones. However, the MSU-MFSD contains only 280 video clips of photo and video attacks from 35 subjects.

1) MSU Unconstrained Smartphone Spoof Attack (USSA) Database: In [1], we collected a replay attack database for smartphones with 465 videos from 155 subjects. Of these 465 videos, 155 were live face videos, and the remaining 310 videos were spoof face videos which were captured by showing the live face videos from the Replay Attack, CASIA FASD and MSU-MFSD databases on a MacBook screen (1280 × 800), and recapturing the face videos using the built-in rear camera of Google Nexus 5 and built-in rear camera of iPhone 6, respectively. Videos were not deliberately captured to include moiré patterns; only a single attempt was made to capture the video. Sample images of these recaptured spoof videos are shown in Fig. 2. A highly desirable property of capturing spoof videos with smartphone devices is that it simulates input videos that may be presented to devices that contain FR systems, such as the Google Nexus 5. The average standoff of the smartphone camera from the screen of the MacBook was 15 cm to ensure that replay videos did not contain the bezels (edges) of the MacBook screen.

In this work, we have significantly increased the number of subjects (1,000+ subjects) as well as the number of live face and spoof images (13,000) in the MSU USSA database. Current public-domain spoof databases often lack diversity in terms of background, illumination, and image quality, and thus do not replicate real application scenarios [45]. The MSU USSA database was specifically created to ensure that it contains a mixture of environments, image qualities, image capture devices and subject diversity. Such a database is essential to obtain generalizable and robust anti-spoofing methods, particularly in face unlock scenarios on smartphones. In real
world scenarios, live face images contain variations in pose, illumination and expression when users use their smartphone throughout the day. However, existing databases contain face images with controlled pose, illumination, and expression variations. The MSU USSA database contains a small percentage of partial-frontal face images as social media sites mainly contain such images and could be used by malicious users to spoof a face recognition system. Running evaluations on a large database of this size will provide statistically significant results for predicting real world performance.

Two versions of the MSU USSA database were created, a 10K (public) and a 13K (private) dataset. The 13K dataset contains images from subjects who withheld consent to share their face images with other researchers as well as images from a private database that we used to supplement the live face images (2,818 additional images in which users withheld consent). However, both the 10K and 13K datasets contain the same number of spoof images (9,120 images). We will report a majority of the results using the public set of the MSU USSA database to allow interested researchers to replicate our findings and further improve face anti-spoofing capabilities.

To create the MSU USSA database, we used a subset (1,000 subjects) of the web faces database collected in [46]. This database contains images of celebrities taken under a variety of backgrounds, illumination conditions and resolutions. We filtered the images to only contain a single frontal facing face image (for smartphone face unlock applications, it is reasonable to expect cooperative user scenario). The other 140 subjects are from the Idiap (50), CASIA FASD (50) and the MSU MFSD (40) public databases. Thus, the new database contains color face images of 1,140 subjects, where the average resolution of the live subject images is 705 × 865.

In order to capture the spoof attacks, we used both the front (1280 × 960) and rear (3264 × 2448) facing cameras on the Google Nexus 5. This allows researchers to study how the quality of the spoof images affects spoof detection performance. Moreover, it allows researchers to examine the images to understand how camera quality affects image quality which in turn affects the presence of image distortion artifacts (i.e., moiré patterns and reflections). Additionally, we captured the spoof attacks to minimize illumination reflections such as the ones shown in Fig. 4. Note the images shown in Fig. 4 are not from the MSU USSA database.

Given that most people have access to a laptop, tablet or smartphone, we captured replay attacks on all three spoof mediums. The spoof attacks are captured by showing the live face image on the screen of one of the spoof mediums and using both the front and rear facing cameras of the Google Nexus 5 to capture the simulated attack. This way, the MSU USSA database was constructed to contain 6,840 images of replay attacks captured using different camera quality and spoof mediums.

In order to capture printed photo attacks, we printed images of all 1,140 subjects using a HP Color Laserjet CP6015xh printer (1200 × 600dpi) on a matte 8.5 ×11 inch white paper. The live subject images were scaled to ensure the image covered as much of the paper as possible while maintaining the original image aspect ratio to minimize distortions. Additionally, we placed the photos in a manner to minimize reflection from ambient lighting inside our laboratory. Both the cameras on the Google Nexus 5 were then used to simulate printed photo attacks to a FR system by capturing a digital image of the printed photo. This way, the MSU USSA database contains 2,280 images of printed photo attacks. Figure 5 shows the setup used to capture both printed photo attacks and replay attacks for the MSU USSA database.

To demonstrate the utility of the collected database for face spoof detection studies, we conducted an experiment using a
Commercial Off-The-Shelf (COTS) face recognition system, which reported promising results in the Face Recognition Vendor Tests 2006 (FRVT)\(^5\). We enrolled the live face images of the 1,140 subjects into a gallery and used the eight spoof images captured for each subject (1,140 subjects) as probe images. In this experiment, at 0.01% FAR, more than 97.7% of the probe images (spoof faces) were successfully matched to their corresponding live face image. This indicates that the MSU USSA database is a realistic database to study face spoof detection as a COTS matcher cannot effectively distinguish between the live and spoof faces.

III. IMAGE DISTORTION ANALYSIS FOR 2D SPOOF FACE IMAGES

Different types of image distortion appear during the re-capture of a face image or video, which generally include (i) surface reflection by the spoof medium, (ii) moiré patterns, (iii) color distortions, and (iv) shape deformations.

A. Spoof Medium Surface Reflection

2D face spoofing attacks are mainly launched by printing a face image or displaying a digital face image or video on a screen. Glossy photo papers and digital screens often generate specular reflections, which lead to reflection distortions in the spoof face images (see Fig. 4). Additionally, both paper and digital screens have different reflective properties than the skin of a face [21], which leads to reflectance differences between live and spoof face images.

B. Color Distortion

Color distribution may change during the capture of a face image, which leads to either reduced color diversity or color cast. For example, while the color distortion of printed attacks is due to the quality of the printer and photo paper, the color distortion of replay attacks is mainly caused by the fidelity and resolution of the screen [20]. Figure 6 shows the color distortion in spoof face images from two subjects in the MSU USSA database.

C. Moiré Pattern

Moiré patterns are an undesired distortion of images caused by an overlap of digital grids [47]. Moiré patterns appear when two or more patterns are overlaid on top of each other, resulting in a third new pattern (Fig. 7 (a)).\(^6\) The display medium of digital devices (laptops, smartphones, and tablets) exhibit a naturally occurring fixed repetitive pattern created by the geometry of color elements that are used for color displays. Therefore, whenever an image of a digital screen is recorded or captured, moiré patterns will most likely present themselves due to the grid overlap between the digital screen and the digital camera. In color printing with CMYK (cyan, yellow, magenta, and black) halftoning model, moiré patterns are often inevitable (Fig. 7 (b)).\(^7\) Moiré patterns are also observed in screen shooting photography (Fig. 7 (c)).\(^8\) The fundamental reason for moiré patterns in screen shooting photography is the spatial frequency differences between the display and the acquisition device. For example, when the image (on the display of a replay device) contains repetitive details that exceed the camera resolution, moiré patterns are observed. While moiré patterns may not appear for replay attacks shown at a distance, replay attacks are typically presented close to a smartphone camera so that the face can be detected. Therefore, moiré patterns can be quite useful in face spoof detection of displayed photo and video replay attacks, particularly for the smartphone unlock application of interest in this paper. [44].

D. Face Shape Deformation

In print attacks, the bending of the photo paper may lead to skewed face shape in the spoof images. Additionally, the

---

\(^5\)http://www.nist.gov/itl/ia/d/frvt-2006.cfm


\(^7\)users.ecs.soton.ac.uk/km/imaging/course/moire.html

\(^8\)blog.ishback.com/?cat=132
viewing direction of the camera will also lead to deformation of the face shape in the spoof images. Figure 8 shows the face shape distortion in spoof face images of print attacks for one of the subjects when displayed on a digital screen.

IV. 2D SPOOF FACE DETECTION ON A ANDROID

In this section, we detail the individual steps of the proposed face spoof detection method for smartphone unlock scenario.

A. Face Detection and Normalization

To detect faces on a smartphone in the input image from its camera, we used the built-in Android face detector. This detector only returns the IPD value and the mid-point of the face if detected; it does not provide coordinates of the left and right eyes. Thus, we used a scale factor between the IPD and the mid-point of a detected face to normalize the face image into a 144 × 120 pixel image. We observed that the Android face detector returns values that can vary greatly leading to inconsistent face cropping even in frames captured only milliseconds apart. This variability in face detection can lead to inaccurate face spoof detection. Hence, for our experiments on a desktop, we use the face detector from the PittPatt face recognition SDK.\(^\text{9}\)

B. Representation

One popular and simple image descriptor for face images is Local Binary Patterns (LBP) [8]. LBP was later generalized to multi-scale LBP (MLBP) which has been shown to perform better than LBP, for example in [48] when matching composite sketches to face photos. This motivated us to conduct experiments to test if MLBP performs better than LBP when detecting for face liveness. Additionally, [49] introduced a new low complexity, effective image descriptor, called Locally Uniform Comparison Image Descriptor (LUCID), which gives comparable results to the well-known SURF descriptor. Since no results have been published on the effectiveness of LUCID on face liveness detection, we conducted experiments to analyze its potential. We also analyzed the SIFT (scale invariant feature transform) feature descriptor as this descriptor is largely invariant to scale, illumination, and local affine distortions [50].

Given the strengths and limitations of individual representation methods (Table I), we choose our feature representation by considering the complementarity between different cues. For example, the color moments based methods depend upon how the image was presented to the FR system when conducting a spoof attack. A digital screen such as a laptop or a smartphone can display millions of colors, thus the color diversity in a spoof image might be very similar to a live face image. However, in printed photo attacks the color diversity is greatly reduced, therefore this feature might be better suited to handle printed photo attacks. The same could be said for blurriness, as high resolution digital screens will display photos with

\(^\text{9}\)PittPatt was acquired by Google in 2011, and the SDK is no longer publicly available.
high definition, but for a printed image, the quality could be degraded. Thus, we hypothesized that a feature level fusion of texture features and image quality features would provide robust spoof detection performance.

Since we are focusing on 2D face spoof detection which contains printed photo, displayed photo, and replayed video attacks, we chose to use the feature representation methods that work for both single face image and multiple video frames. Table III provides a summary of various feature extraction methods that we considered.

Given the performance of individual features and requirement of a fast response spoof detection system on smartphones, we chose to use a fusion of LBP (effective for face texture analysis) and color moments (effective for image quality analysis). Color moments tend to highlight the differences in color distribution in live face images compared to spoof attacks. To calculate these color moments, we first convert an RGB image into the HSV (Hue, Saturation, and Value) space and then compute the mean, deviation and skewness of each channel as a color feature [20]. To extract the face texture features, we calculate LBP$_{P,R}$, with parameter values $P = 8$, and $R = 1$, by dividing the image into $32 \times 32$ patches with 16 pixels overlap. Parameter $P$ defines the quantization of the angular space and parameter $R$ defines the spatial resolution of the operator (radii). The LBP features from individual patches are concatenated together to construct a feature vector with 4,248 dimensions. As shown in Fig. 9, the LBP feature descriptor can capture patterns that appear in spoof imagery quite effectively.

The proposed complementary feature representation is effective in detecting individual image distortion artifacts in spoof face images (summarized in Section III), particularly under cross-database testing scenarios. Specifically, while texture analysis of the proposed approach is effective in capturing surface reflection, moiré pattern, and shape deformation, image quality analysis of the proposed approach is effective in capturing color distortion and surface reflection. Additionally, this 4,263-dimensional feature vector can be computed very efficiently, 0.021 sec. per face image, on average (about 47 FPS). Reported times are profiled with a Matlab implementation on a Windows 7 platform with Intel Core 2 quad 3.0 GHz CPU and 8GB RAM.

While the previous work in [20] also studied image quality features for spoof detection, in this work we propose a complementary feature representation by considering both image quality and face texture features. Such a representation leads to more robust performance on the challenging scenarios (such as the CASIA FASD database). Additionally, we build a large face spoof database with over than 1,000 subjects and 13,000 images for robust training of spoof detector and evaluation.

C. Multi-frame Voting

We perform face spoof detection on smartphones by capturing a sequence of three face image frames. We enforce a 200 millisecond separation between the successive image captures to allow for the motion of a subject’s hand holding the device to introduce subtle changes in the images captured.

Given the feature vectors extracted from the training images, we train a SVM classifier with an RBF kernel (using optimized parameters) to distinguish between live and spoof faces. If two or more frames within the three frames in a session are classified as live faces, then a given session will be classified as live, otherwise a spoof (majority voting). Using input from multiple frames allows us to stabilize the decision for a session. While more frames may further improve the performance, we use three frames to ensure the proposed approach can run efficiently on smartphones.

D. Reject Option

We observed that most malicious users tend to hold the spoof medium (smartphone or printed photo) at a certain distance to a smartphone camera when they are trying to spoof FR systems. They do this as they believe this will lead to higher quality face images being captured. Additionally, to hide the evidence of a spoof attack (bezels of a digital device and boundary of a printed photo) a malicious user may need to hold the spoof medium as close as possible to the FR system. An experiment conducted on 10 subjects showed that malicious users indeed tend to hold the spoof medium as close as possible to the FR system. This motivated us to utilize a threshold on IPD to reject an image.

In order to find an acceptable range of IPD, we conducted experiments using 20 subjects, where we asked the users to take 10 pictures of themselves using a Google Nexus 5. Subjects that were used for this study had arms of varying lengths. The subjects were instructed to hold a smartphone as they would during normal usage and to capture a number of selfie pictures. Using these 200 images, we determined the typical IPD values under normal smartphone use. The average IPD of live faces (captured by the front facing camera of a smartphone) is $\mu_{IPD} = 28.8\%$ of the image width (720 pixels), and the standard deviation of IPD is $\sigma_{IPD} = 3.6\%$ of image width. Based on these statistics, we reject faces that are either too small (faces that are very far from the smartphone camera) or too large (faces that are very close to the smartphone camera) by using

$$r_{IPD}(d,a) = |d - \mu_{IPD}| \leq a \cdot \sigma_{IPD},$$

By setting $a = 2$, about 95% of the input face images to the smartphone camera are accepted and submitted to the spoof detection system.

Additionally, we define another reject option based on the detection of bezels of the spoof medium being used. This is done by detecting black stripes along the left and right sides (bezels) of the image as shown in Fig. 3, when the whole image is used. These stripes quickly allow us to detect spoof attacks, as these black stripes will only appear on digital screens such as on laptops, smartphones and tablets.

The bezel detection algorithm looks for areas in which the pixel intensity values remain fairly consistent along the top,
Fig. 10. The proposed spoof detection method with reject options and complementary feature representation (face texture: LBP, and image quality: Color Moments). An input image frame will be skipped, if face detection fails.

Fig. 11. Examples of inputs that were rejected using the proposed reject option: (a) IPD value below the lower threshold, (b) IPD value above the upper threshold, and (c) detected bezels along the top and left side of input image.

bottom, right, and left edges of an image. Bezels tend to be uniform in color (black) along the borders and thus the pixel intensity values remain fairly consistent. On the other hand, in live subject images, the pixel intensity values in the background tend to vary significantly. We analyze columns of 60 consecutive pixels for bezel detection for the left and right side of an image and rows of 50 pixels for the top and bottom of an image given a normalized image of 144 × 120 pixels. We iteratively analyze up to 10 different areas by moving the areas in question closer to the center by 3 pixels at a time until we reach a max offset of 30 pixels from the edge. The system will report a bezel detection if any of these areas satisfy the following constraint

\[
t(\mu, \sigma) = \begin{cases} 
1 & \text{if } \mu < 5, \ \sigma < 5 \\
1 & \text{if } \mu > 220, \ \sigma < 5 \\
0 & \text{otherwise}
\end{cases}
\]

(2)

where \(\mu\) and \(\sigma\) are defined as the average pixel intensity value and the standard deviation for the area in question. The parameter values were determined empirically based on the performance on individual scenarios.

Biometric systems with reject options are not new [51], but studies of reject options for face spoof detection are limited. Using the two reject options described above greatly helps in detecting spoof attacks using minimal processing time. The combination of restricting the IPD of a subject and detecting the bezels of an input image reduces the number of images that are processed using the proposed spoof detection method. This is due to the fact that when replay attacks are fabricated, the restriction on the IPD leads to capture of images that often contain the bezels of the spoof medium. Thus, the reject options help in reducing the number of false accepts in our system. Moreover, due to the update to Face Unlock with the release of Android 5.0, users no longer can view what the FR system is capturing. Therefore, malicious users no longer can ensure input to a FR system is free of any bezels.

It should be noted that printed photo attacks may evade the bezel detection if a malicious user cuts the image to remove the bezels. However, the restriction of the IPD may still help detect the presence of a printed-photo attack. The goal is to reject the “easy cases” of spoof attacks using minimal processing time. Figure 10 shows the system diagram of the proposed method and how the reject option fits into our overall spoof face detection system. Figure 11 shows a couple of examples of input face images that are rejected by the proposed method.

E. Prototype System on a Smartphone

We implement a prototype system of the proposed approach on a Nexus 5 with API level-21 support from Android v5.1. A minor change over the proposed method in the prototype system on desktop is that now we use Android face detector instead of the PittPatt face detector. This necessitated a retraining of our face spoof detection model by utilizing the Android face detector to detect individual faces in the training dataset, and retraining the face spoof detector using the same method described in Sections IV.B-IV.D.

V. EXPERIMENTAL RESULTS

We perform face spoof detection experiments using the MSU USSA database, and the Idiap Replay-Attack, CASIA FASD, MSU-MFSD and RAFS spoof face databases. We study the influences of a number of factors (e.g., image acquisition device, image region, IPD, and database size) to the proposed face spoof detection approach. The proposed approach is compared with state of the art methods in both cross-database and intra-database testing scenarios. Besides performing evaluations using the original testing protocol of each database, we also design a protocol for the scenario of face unlock on smartphones. The subject IDs used in each
TABLE IV

PERFORMANCE OF FACE SPOOF DETECTION USING FACE IMAGES CAPTURED WITH FRONT AND REAR CAMERAS OF A NEXUS 5 PHONE.

<table>
<thead>
<tr>
<th>Training Set</th>
<th>Testing Set</th>
<th>FAR (%)</th>
<th>FRR (%)</th>
<th>HTER (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rear Camera</td>
<td>Front Camera</td>
<td>67.80</td>
<td>0.44</td>
<td>34.14 ± 1.46</td>
</tr>
<tr>
<td>Front Camera</td>
<td>Rear Camera</td>
<td>46.54</td>
<td>1.30</td>
<td>23.92 ± 1.79</td>
</tr>
<tr>
<td>Rear Camera</td>
<td>Rear Camera</td>
<td>7.83</td>
<td>0.44</td>
<td>4.14 ± 0.55</td>
</tr>
<tr>
<td>Front Camera</td>
<td>Front Camera</td>
<td>9.31</td>
<td>1.30</td>
<td>5.31 ± 0.51</td>
</tr>
</tbody>
</table>

†A five-fold cross-validation protocol is used. Rear Camera signifies spoof image captured by the rear facing camera on the Nexus 5 and the Front Camera signifies spoof images captured by the front facing camera. FRR is the false rejection rate of live face images, and FAR is the false acceptance rate of spoof face images.

A fold of the designed five-fold subject-exclusive cross validation protocol for the MSU USSA database will be included in our public release. Unless otherwise stated, the experiments are conducted using the public set of the MSU USSA database.

A. Influence of Image Acquisition Device

Table IV shows the effects on face spoof detection when cameras of different specifications are used to capture the training and testing face images. When face images from the training and testing sets are captured using cameras of different specifications, the HTER is larger than the HTER when the training and testing face images are captured using the same camera. See Table IV. This performance gap is mainly due to the fact that moiré patterns do not appear when the frontal facing camera is used as it lacks autofocus capabilities.12 Hence, this leads to a dramatic increase in the FAR while maintaining the FRR. To close this performance gap, the training set used to learn face spoof detection models should include a wide variety of image acquisition devices for both live and spoof face images. Therefore, the MSU USSA database should help to learn better face spoof detection models, as it contains both live and spoof face images captured using several different cameras.

B. Influence of Different Image Regions

We study the effect of different image regions (i.e. whole image, detected face image, and bottom half of a face) on spoof detection performance using the MSU USSA database (see Fig. 13). To our surprise, at 0.01% FAR the performance when using the whole image to train face spoof detection models is better than when using the detected face region. This result seems to be counter to the prevailing wisdom that the background area of a face contains noise which may degrade performance. However, after further examination, we realize that the trained model is tuned to detect the black stripes along the left and right sides (bezels) of the image, when the whole image is used as mentioned in Section IV.D. Thus, when we consider the whole image, only images that did not contain any black strips along the edges were misclassified. Therefore, face spoof detection models specifically trained with whole images can efficiently detect printed photo and replayed video attacks, which often have black stripes due to the limited sizes of the photograph paper and screen. However, in more challenging scenarios, e.g., when no black stripes appear in spoof face images (particularly when a malicious user intentionally prevents the paper or screen boundary appearing in the camera’s field of view), experiments in [1] showed that using the detected facial region provides better performance than using the whole image.

C. Influence of Color Channel

We analyze the performance of the proposed face spoof detection method by using LBP features extracted from the grayscale, red, green and blue channels of the detected face images from the MSU USSA database (see Fig. 14). We only extracted LBP features as the color moment features require an RGB image. Figure 15 shows that different color channels highlight varying amounts of texture in an image, hence leading to performance differences. The red channel gives better performance than the other color channels. Apparently, texture component that can distinguish between live and spoof faces has higher contrast in the red channel of a face image.

12 Most smartphones being released now have autofocus capabilities in the front camera as well (HTC Desire Eye, ZTE Blade S7).
D. Influence of IPD

Given that most published methods extract features from the detected facial region, we analyze how the IPD affects the face spoof detection performance on the MSU USSA database. Given the cropped face images of a fixed size \((144 \times 120)\), we vary the cropping of the facial region by altering the IPD (i.e. 40, 50, 60 and 70 pixels). Figure 16 shows the normalized face image of a subject when cropping images using different IPD.

As shown by the ROC curves in Fig. 17, using an IPD of 60 or 70 pixels when cropping a face leads to better performance than using an IPD of 40 or 50. This is due to the fact that cropping a face to have an IPD of 60 or 70 pixels removes most of the background area while retaining as large of the facial region as possible. Removing the background eliminates the background clutter from a normalized face image while a large facial region retains more distinctive features for classification of live and spoof face images. When reporting results for all other experiments, we normalize faces images to an IPD of 60 pixels as using an IPD of 60 pixels obtains a higher true accept rate at low false accept rates.

E. Influence of Database Size

Most of the available public domain face spoof databases contain no more than 50 subjects. Therefore, we study how the number of subjects in the training set affects spoof detection performance using the MSU USSA database. Table V shows that using a larger training set from the MSU USSA database significantly improves the cross-database performance under the smartphone protocol when testing on the Idiap Replay-Attack, CASIA FASD and MSU-MFSD databases. When using only 1,000 spoof images to train our classifier, the spoof detection performance significantly degrades compared to when we used all 8,000 spoof images. In fact, as we use more and more spoof images to train the SVM classifier, the performance keeps increasing. On the public-domain databases such as Idiap replay-attack and CASIA FASD, we also noticed such trend that utilizing more frames from each video for training leads to better cross-database performance on a completely different database. The above results show that increasing the number of training face images to cover more diversities, from individual subjects to image acquisition devices, helps to learn more robust classifiers. Thus, larger databases such as the MSU USSA database will be very helpful in advancing solutions to the face spoof detection problem.

F. Intra-database Testing

We evaluate the proposed approach under the intra-database testing scenarios on the newly created MSU USSA, Idiap Replay-Attack, CASIA FASD, and MSU-MFSD databases. Example images of subjects from these databases are shown in Fig. 2. We perform these tests using the protocols specified in [7], [20], and [40] as well as a protocol we define to simulate smartphone spoof attacks. In order to make these databases more compatible to spoof attacks on smartphones, we used the spoof videos generated for the Idiap Replay-Attack, CASIA FASD and MSU-MFSD databases from the RAFS database (smartphone protocol). The RAFS database recaptured the spoof videos for these three databases using a smartphone compared to the low resolution webcams and DSLR cameras used in the original spoof videos. Note, we
did not leverage the reject option for any of these databases, as these databases were collected in a controlled manner to limit bezels in the videos and constraint the IPD.

On the Idiap Replay-Attack database using the original protocol, the proposed approach achieves 14.6% HTER which is larger than [20] (7.41%) and [14] (2.9%) but lower than [19] (15.2%). On the CASIA FASD database using the original protocol, the proposed approach achieves 5.88% EER, which is smaller than the EER reported in several other publications (6.20% [14], 7.2% [15], 12.9% [20], 14.0% [23]). On the MSU-MFSD using the original protocol, the proposed approach achieves 8.41% EER which is slightly larger than the approach in [20] (5.82%). However, under the smartphone protocol for face unlock, the proposed approach achieves very promising results (0% HTER on Idiap Replay-Attack, 1.67% EER on CASIA FASD, and 2.67% EER on MSU-MFSD databases). In all these experiments, the proposed method achieves comparable performance to the state of the art methods under the original testing protocols of Idiap Replay-Attack, CASIA FASD, and MSU-MFSD databases. In all these experiments, the proposed method achieves comparable performance to the state of the art methods under the original testing protocols of Idiap Replay-Attack, CASIA FASD, and MSU-MFSD databases but achieves much better performance using the smartphone protocol. The main reason for the difference in performance under the two protocols is that discriminative cues (moire patterns, color diversity, etc.) between live and spoof subject videos are more prevalent in spoof videos captured by smartphones; DSLR cameras contain advance features (specialized lens, anti-aliasing filters) to normalize such image distortions. Thus, spoof videos in the smartphone protocol more closely represent input that a face unlock system on a smartphone would receive compared to DSLR cameras.

The protocol we used for intra-database testing on the MSU USSA database was a subject-exclusive five-fold cross validation, where the subjects were randomly split into 5 folds. We will share the subjects’ ID list used in each fold of the 5-fold protocol so that interested researchers can replicate our results. As shown in Table III, the proposed method achieved EER of 3.51% and 3.84% on the private and public sets, respectively. The ROC curves for these tests are shown in Fig. 18. Additionally, the proposed approach achieved EER of 2.87% and 4.06% for photos displayed on screen (replay attack) and photos printed on paper, respectively, when using the public set of the MSU USSA database. The results show that these two types of photo attacks can be detected with similar accuracies. The ROC curve for this test is also shown in Fig. 18.

G. Cross-database Testing

It is now generally accepted that intra-database testing (where training and testing images, while distinct, are captured in the same environment and possibly of the same subjects) does not represent real world scenarios, and it lacks generalization ability [11]. Therefore, we also evaluated the proposed approach under cross-database testing scenarios. The cross-database protocol performance is evaluated by training an anti-spoofing method on database A and testing it on a different database B. We used the public set of MSU USSA database to train a face spoof detection model based on the proposed method and then test it on the MSU-MFSD, Replay-Attack and CASIA FASD databases based on the smartphone protocol. To avoid any bias, we removed the overlapping subjects (40 from MSU-MFSD, 50 from Replay-Attack, and 50 from CASIA FASD) that appear in both the MSU USSA database and the testing databases. As shown in Table V, the proposed approach achieves 9.27%, 3.50%, and 2.00% HTERs on the MSU-MFSD, Idiap Replay-Attack and CASIA FASD databases, respectively, using the smartphone protocol. These results support the proposed claim that our method has good generalization ability as it reports fairly low HTERs in the challenging cross-database testing protocol. Additionally, this underscores the fact that our method can differentiate live subject images vs. spoof images using image quality features such as moiré patterns and color moments.

Examples of correct classifications and misclassifications by the proposed approach on cross-database testing are shown in Fig. 19. No examples of false reject of live face images...
are reported by the proposed approach because in all three experiments, the false reject rate is 0. We find that many of the errors can be attributed to poor image quality such as over saturation of images and color distribution which are not represented in our training dataset. Additionally, some of the errors are caused by motion blur and incorrect face cropping due to dark skin.

In Table I, we also provide the best cross-database performance achieved on the MSU-MFSD, Idiap Replay-Attack, and CASIA FASD databases utilizing the original protocol for the databases (original spoof videos), where we trained a model using one of the three databases and tested the model on the other two databases. Under this protocol, the cross-database spoof detection performance degrades (26.7% vs. 9.27% HTER on MSU-MFSD, 29.3% vs. 3.50% HTER on Idiap Replay Attack and 35.4% vs. 2.00% HTER on CASIA FASD). However, as we summarized in Section II.B, these databases (captured using webcam or DSLR) do not replicate smartphone unlock scenarios. Thus, we want to emphasize the performance of the smartphone protocol as the application of interest in this paper is spoof detection for smartphones.

TABLE VI

BEZEL DETECTION PERFORMANCE ON NINE DISTINCT IMAGE SETS IN THE MSU USSA DATABASE. THIS TABLE SHOWS THE PERCENTAGE OF IMAGES IN WHICH THE PROPOSED BEZEL DETECTOR DETECTED A BEZEL. FRONT AND REAR SIGNIFY THE CAMERA OF THE NEXUS 5 USED TO CAPTURE THE SPOOF IMAGES ON THE SPECIFIED SPOOF MEDIUM.

<table>
<thead>
<tr>
<th>Image Set</th>
<th>Percentage of Images</th>
</tr>
</thead>
<tbody>
<tr>
<td>Live</td>
<td>12.3%</td>
</tr>
<tr>
<td>Front MacBook</td>
<td>86.5%</td>
</tr>
<tr>
<td>Rear MacBook</td>
<td>83.9%</td>
</tr>
<tr>
<td>Front Nexus</td>
<td>95.0%</td>
</tr>
<tr>
<td>Rear Nexus</td>
<td>88.7%</td>
</tr>
<tr>
<td>Front Tablet</td>
<td>99.9%</td>
</tr>
<tr>
<td>Rear Tablet</td>
<td>82.6%</td>
</tr>
<tr>
<td>Front Printed Photo</td>
<td>25.6%</td>
</tr>
<tr>
<td>Rear Printed Photo</td>
<td>53.0%</td>
</tr>
</tbody>
</table>

H. Bezel Detection Performance

We evaluated the performance of our bezel detection algorithm on the MSU USSA database. Table VI shows the results of our bezel detector on the 9-image sets in the database (1 live face image set and 8 spoof face image sets). The reason why 12.3% of live face images are detected to have a bezel is because many of these images were captured against a pure white or black background using professional grade cameras as shown in Fig. 20(a). When we removed such images from the live image set, only 1.4% of live face images had a falsely detected bezel. Moreover, in Section V.I we show that our bezel detector has a low (but non-zero) FAR when implemented on a smartphone.

For the 8 spoof image sets (4 spoof mediums × 2 cameras), a bezel was detected for many of these 9,120 images. However, some of these spoof images did not contain any bezel. The spoof face images captured by the front facing camera from the Tablet screen, always contained a bezel due to our camera positioning, and on this image set the bezel detector detected bezels with 99.9% accuracy. This shows that our algorithm can detect bezels with high accuracy. Additionally, if we removed non-bezel images from the printed photo attacks captured by the front and rear facing cameras of the smartphone, bezels were correctly detected with 97.7% accuracy for the rear camera image set and with 84.9% accuracy for the front
camera image set. Thus, our reject option based on bezel detection is effective in identifying spoof input to a FR system.

I. Performance Evaluation on Smartphones

We evaluate the performance of our Android application by asking 20 subjects to use the app for routine smartphone unlock. The spoof detector application was loaded onto a Google Nexus 5 and a HTC Desire Eye (see GUI in Fig. 21). These subjects were chosen to make sure that the test set was diverse in terms of race, age, sex and facial hair style. The face spoof detector was trained on a desktop using the MSU USSA database.

One set of experiments was designed to determine whether our application could successfully detect live faces. These tests were conducted in various illumination conditions such as a dark hallway, sunny outside environment, and an indoor apartment setting with a large window. The users were instructed to hold the phone at different arm lengths and to move around in their environment to introduce illumination variations. They were then instructed to periodically press the “verify” button on the application so that the result of face liveness detection could be automatically recorded. For each subject, five verification tests were conducted. Among the 100 live face attempts (5 per subject), our Android application successfully accepted 96 faces (96.0% accuracy) on the Google Nexus 5 and 94 faces (94.0% accuracy) on the HTC Desire Eye. In these tests, we did have the reject option turned on. We only encountered a single case in which the live subject was rejected by the bezel detection in all 5 verification attempts. These false rejections occurred due to the fact the subject was wearing a solid black shirt and that the test was conducted in a dark corner of the room. When we repeated the tests in the center of the room, where the illumination conditions were better, the bezel detection did not falsely reject the live user.

Additionally, we conducted experiments to determine whether the application could effectively detect spoof face access. We asked the participating subjects to capture selfie images, which we would use later to launch spoof face attacks. For spoof attacks, the selfie images were displayed on an iPhone 6 and an Apple MacBook Pro laptop with retina display. Again, we did five tests per spoof medium. Among the 200 spoof face accesses, our Android application on the Google Nexus 5 correctly rejected 155 spoof faces (77.5% accuracy) and 157 spoof faces (78.5% accuracy) when the MacBook Pro laptop and iPhone 6 were used as the spoof medium, respectively. On the HTC Desire Eye, our application correctly rejected 136 spoof faces (68.0% accuracy) and 162 spoof faces (81.0% accuracy) when the MacBook Pro laptop and iPhone 6 were used as the spoof medium, respectively. The above spoof face detection results were recorded by turning off our reject option. If we use the reject option, numerous inputs to the FR system were rejected due to the detection of a bezel and the IPD constraint. The spoof detection performance of our application on both Google Nexus 5 and HTC Desire Eye reached the high 90% range if reject option was utilized.

The performance achieved in a cross-database testing scenario as reported in the literature, is not very good compared to intra-database testing (average HTER of 47.7% reported in [11] and 38.97% reported in [23]). However, the proposed face spoof detection system running on smartphone is able to achieve accuracies in the 80% range. Moreover, our results showcase the fact that performance obtained on laboratory collected databases tend not to reflect real world performance when users actually leverage spoof detection applications.

The results above also show that while the face spoof detection system was trained on the MSU USSA database, it still runs smoothly on HTC Desire Eye smartphone which has a completely different camera than the cameras used in collecting the MSU USSA database. These results show that the proposed approach and a large training database, namely MSU USSA, do not lead to a biased system; it does not simply detect different sensors and shows it generalizes well to different image acquisition devices.

For the incorrect classifications in live face unlock test, poor illumination condition is the main reason for failure, particularly dim light and yellow light. The main reason for the false acceptances of spoof face accesses is the lack of moiré patterns which are due to the occasional slow autofocus capability of the smartphone cameras.

J. Moiré Pattern Detection on A Smartphone

Given an input face image, our method will classify it as a spoof if moiré patterns are detected. As we discussed in Sec. III. C and in [1], the presence of moiré patterns provides evidence of displayed photo and video replay attacks launched using a digital screen. The method used for moiré pattern detection here is the same as in our earlier conference paper [1]. As shown in [1], moiré patterns can be well represented using LBP descriptor, and thus moiré pattern detection method is naturally embedded in the proposed approach. It can also be used as a pre-filtering stage similar to the reject option.

To verify that our Android application is effective in detecting moiré patterns, we tested it on non-face images such as solid color images, outdoor images, and wallpapers containing cars. For each of these images (three such images shown in the top row of Fig. 22), five verification tests were conducted. Among the 75 spoof attempts, our Android application
correctly rejected 65 of them (86.7% accuracy) when using a MacBook Pro laptop to display the non-face images. This experiment shows that the proposed method still performs well for detecting displayed photo and video replay attacks, even if the face detection does not give accurate face detection results.

K. Running Time and Memory Requirement on Smartphones

The Android spoof detection application must provide fast response to the users. The current implementation takes 0.02 seconds for classification and 1.65 seconds to extract features from a single image frame (144 × 120) for a total time of 1.67 seconds. However, using three frames to make a decision leads to only a marginal increase in the total time to 1.95 seconds because of our multithreaded implementation on Android. Reported times are profiled on a Google Nexus 5 smartphone with 2GB of RAM and Quad-core 2.3 GHz Krait 400 CPU running native Android 5.0 ROM. As a comparison, the proposed approach takes 0.03 seconds on a desktop (see Section IV.B for desktop specification) for feature extraction and classification of a single frame. Our goal is to bring down the total time to less than 1 sec.

On the Google Nexus 5, our application utilizes 53 MB of RAM, a minuscule amount compared to the gigabytes of RAM available on smartphones today. Of the 53 MB, 15 megabytes are allocated for the SVM model file that was trained on the desktop. This model file contains 6,248 support vectors for the RBF SVM classifier.

VI. SUMMARY AND CONCLUSIONS

Spoofing attacks can be easily launched against face recognition systems due to the low cost of obtaining printed photos or video replays. In order to address the problem of face spoofing on smartphones, we propose an efficient detection approach based on the analysis of image distortions in 2D spoof face images and the complementarity of individual cues (LBP and color moments). We also collected a large database, called the MSU Unconstrained Smartphone Spoof Attack (MSU USSA), that contains replay and printed photo attacks captured by different smartphone cameras. Experimental evaluations show that a large database is essential to learn robust face spoofing detection models, particularly under cross-database testing scenarios. Moreover, we show that IPD based face normalization followed by feature extraction from the red color channel rather than a grayscale image increases spoof detection performance. Additionally, we propose a simple but efficient reject option for face images based on IPD constraint and bezel detection. The proposed spoof detection method was implemented on two Android smartphones (Google Nexus 5 and HTC Desire Eye), and the proposed approach can perform face spoof detection efficiently on commodity smartphones. We plan to make use of the temporal and contextual information included in multiple video frames to build more robust face spoof detection models. Additionally, we plan to combine the proposed method with movement cues (e.g., eye-blink) to further improve face spoof detection performance.

VI. SUMMARY AND CONCLUSIONS

Spoofing attacks can be easily launched against face recognition systems due to the low cost of obtaining printed photos or video replays. In order to address the problem of face spoofing on smartphones, we propose an efficient detection approach based on the analysis of image distortions in 2D spoof face images and the complementarity of individual cues (LBP and color moments). We also collected a large database, called the MSU Unconstrained Smartphone Spoof Attack (MSU USSA), that contains replay and printed photo attacks captured by different smartphone cameras. Experimental evaluations show that a large database is essential to learn robust face spoofing detection models, particularly under cross-database testing scenarios. Moreover, we show that IPD based face normalization followed by feature extraction from the red color channel rather than a grayscale image increases spoof detection performance. Additionally, we propose a simple but efficient reject option for face images based on IPD constraint and bezel detection. The proposed spoof detection method was implemented on two Android smartphones (Google Nexus 5 and HTC Desire Eye), and the proposed approach can perform face spoof detection efficiently on commodity smartphones. We plan to make use of the temporal and contextual information included in multiple video frames to build more robust face spoof detection models. Additionally, we plan to combine the proposed method with movement cues (e.g., eye-blink) to further improve face spoof detection performance.

VI. SUMMARY AND CONCLUSIONS

Spoofing attacks can be easily launched against face recognition systems due to the low cost of obtaining printed photos or video replays. In order to address the problem of face spoofing on smartphones, we propose an efficient detection approach based on the analysis of image distortions in 2D spoof face images and the complementarity of individual cues (LBP and color moments). We also collected a large database, called the MSU Unconstrained Smartphone Spoof Attack (MSU USSA), that contains replay and printed photo attacks captured by different smartphone cameras. Experimental evaluations show that a large database is essential to learn robust face spoofing detection models, particularly under cross-database testing scenarios. Moreover, we show that IPD based face normalization followed by feature extraction from the red color channel rather than a grayscale image increases spoof detection performance. Additionally, we propose a simple but efficient reject option for face images based on IPD constraint and bezel detection. The proposed spoof detection method was implemented on two Android smartphones (Google Nexus 5 and HTC Desire Eye), and the proposed approach can perform face spoof detection efficiently on commodity smartphones. We plan to make use of the temporal and contextual information included in multiple video frames to build more robust face spoof detection models. Additionally, we plan to combine the proposed method with movement cues (e.g., eye-blink) to further improve face spoof detection performance.

VI. SUMMARY AND CONCLUSIONS

Spoofing attacks can be easily launched against face recognition systems due to the low cost of obtaining printed photos or video replays. In order to address the problem of face spoofing on smartphones, we propose an efficient detection approach based on the analysis of image distortions in 2D spoof face images and the complementarity of individual cues (LBP and color moments). We also collected a large database, called the MSU Unconstrained Smartphone Spoof Attack (MSU USSA), that contains replay and printed photo attacks captured by different smartphone cameras. Experimental evaluations show that a large database is essential to learn robust face spoofing detection models, particularly under cross-database testing scenarios. Moreover, we show that IPD based face normalization followed by feature extraction from the red color channel rather than a grayscale image increases spoof detection performance. Additionally, we propose a simple but efficient reject option for face images based on IPD constraint and bezel detection. The proposed spoof detection method was implemented on two Android smartphones (Google Nexus 5 and HTC Desire Eye), and the proposed approach can perform face spoof detection efficiently on commodity smartphones. We plan to make use of the temporal and contextual information included in multiple video frames to build more robust face spoof detection models. Additionally, we plan to combine the proposed method with movement cues (e.g., eye-blink) to further improve face spoof detection performance.

VI. SUMMARY AND CONCLUSIONS

Spoofing attacks can be easily launched against face recognition systems due to the low cost of obtaining printed photos or video replays. In order to address the problem of face spoofing on smartphones, we propose an efficient detection approach based on the analysis of image distortions in 2D spoof face images and the complementarity of individual cues (LBP and color moments). We also collected a large database, called the MSU Unconstrained Smartphone Spoof Attack (MSU USSA), that contains replay and printed photo attacks captured by different smartphone cameras. Experimental evaluations show that a large database is essential to learn robust face spoofing detection models, particularly under cross-database testing scenarios. Moreover, we show that IPD based face normalization followed by feature extraction from the red color channel rather than a grayscale image increases spoof detection performance. Additionally, we propose a simple but efficient reject option for face images based on IPD constraint and bezel detection. The proposed spoof detection method was implemented on two Android smartphones (Google Nexus 5 and HTC Desire Eye), and the proposed approach can perform face spoof detection efficiently on commodity smartphones. We plan to make use of the temporal and contextual information included in multiple video frames to build more robust face spoof detection models. Additionally, we plan to combine the proposed method with movement cues (e.g., eye-blink) to further improve face spoof detection performance.


