Seeing Your Face Is Not Enough: An Inertial Sensor-Based Liveness Detection for Face Authentication

Yan Li, Yingjiu Li, Qiang Yan, Hancong Kong, Robert H. Deng
School of Information Systems, Singapore Management University
{yan.li.2009, yjli, qiang.yan.2008, hckong.2014, robertdeng}@smu.edu.sg

ABSTRACT

Leveraging built-in cameras on smartphones and tablets, face authentication provides an attractive alternative of legacy passwords due to its memory-less authentication process. However, it has an intrinsic vulnerability against the media-based facial forgery (MFF) where adversaries use photos/videos containing victims' faces to circumvent face authentication systems. In this paper, we propose FaceLive, a practical and robust liveness detection mechanism to strengthen the face authentication on mobile devices in fighting the MFF-based attacks. FaceLive detects the MFF-based attacks by measuring the consistency between device movement data from the inertial sensors and the head pose changes from the facial video captured by built-in camera. FaceLive is practical in the sense that it does not require any additional hardware but a generic front-facing camera, an accelerometer, and a gyroscope, which are pervasively available on today's mobile devices. FaceLive is robust to complex lighting conditions, which may introduce illuminations and lead to low accuracy in detecting important facial landmarks; it is also robust to a range of cumulative errors in detecting head pose changes during face authentication.

Categories and Subject Descriptors

D.4.6 [**OPERATING SYSTEMS**]: Security and Protection—*Authentication*; K.6.5 [**MANAGEMENT OF COMPUTING AND INFORMATION SYSTEMS**]: Security and Protection—*Authentication*

Keywords

Liveness detection; face authentication; media-based facial forgery

1. INTRODUCTION

Face authentication has been commonly used in commercial products of mobile devices, such as smartphones and tablets. Face authentication systems. It provides a potential alternative to legacy passwords as it requires no user memory while providing a higher entropy for identifying users [34]. Unfortunately, most existing face authentication systems, including Face Unlock [16], Facelock Pro [11], and Visidon [48], have an intrinsic vulnerability against the *media-based facial forgery* (MFF) where an adversary forges or replays a photo/video containing a victim's face. Prior research has shown that 53% of facial photos in online social networks, such as Facebook and Google+, can be simply used to spoof such face authentication systems [28]. The MFF-based attacks pose a severe threat to the existing face authentication systems.

In order to defend against the MFF-based attacks, liveness detection is required to distinguish between the legitimate face biometrics of live users and the forged face biometrics [3, 24, 26, 36]. Simple liveness detection techniques have been proposed for detecting the *photo-based attacks* where an adversary replays a facial photo. For example, the eye blink based approach and the head rotation based approach require users to blink their eyes or rotate their heads [24, 36]. Such liveness detection approaches are still subject to *video-based attacks* where a more powerful adversary replays pre-recorded face videos or multiple modified images containing the required motions [38].

More advanced liveness detection techniques are proposed against both photo-based attacks and video-based attacks. For example, the facial thermogram based approach analyzes an additional input of thermogram data from an infrared camera [15]. The optical flow analysis based approach examines liveness clues in a high-quality input of images/videos captured under ideal indoor lighting conditions [24]. A recent research work thwarts the MFF-based attacks by accurately detecting a clear edge of nose which requires controlled lighting and clear appearance of nose [8]. However, the required inputs of these approaches may not be easy to obtain on mobile devices in practice due to real-world challenges such as *limited hardware capability* and *varied usage environments*. Device manufacturers are reluctant to add extra hardware features like infrared camera if they are not driven by strong business demand. Meanwhile, the real non-ideal lighting conditions in varied usage environments may lead to low quality of input images/videos due to noisy pixels, loss of pixels, illumination, etc. The real lighting may also diminish the geometric features of important facial landmarks such as nose and eyes and make it difficult to detect such facial landmarks in face authentication.

To address these challenges, we propose FaceLive, an inertial sensor-based liveness detection mechanism for face authentication on mobile devices. FaceLive can detect the MFF-based attacks including both photo-based attacks and video-based attacks. It does not require any additional hardware but a generic front-facing camera, an accelerometer, and a gyroscope which are commonly available on mobile devices. To thwart the MFF-based attacks, FaceLive detects 3D characteristics of a live user's face by measuring the consistency between estimated head pose changes from the captured face video and the estimated movements from the inertial

CCS'15, October 12–16, 2015, Denver, CO, USA.
© 2015 ACM. ISBN 978-1-4503-3832-5/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2810103.2813612.

sensors. In order to verify that the input is indeed from a live user, FaceLive requires the user to simply hold and move a mobile device in front of his/her face over a short distance while the front-facing camera on the device captures the video about the user's face and the inertial sensors record the motion data about the device simultaneously. A live user is detected if the changes of head poses in the facial video are consistent with the device movements.

We conduct a user study to validate the proposed liveness detection mechanism. We collect real video data and inertial sensor data from both legitimate authentication requests and MFF-based attacks. We measure the Equal Error Rate (EER) of FaceLive in varied scenarios, including unsuccessful detection of partial facial landmarks in real-world lighting conditions, and inaccurate estimation of head pose changes in liveness detection. The experimental results indicate that FaceLive can detect the MFF-based attacks effectively with low EER rates in all tested scenarios representing varied environments, which shows the practicability and robustness of FaceLive.

2. RELATED WORK

Various liveness detection techniques for face authentication have been proposed in the literature. We summarize them according to the types of their liveness indicators, which include multimodal, texture pattern, 3D face, and real-time response.

Multimodal based liveness detection approaches take face biometrics and other biometrics into account in user authentication. Rowe et al. proposed a multimodal based technique which requires a camera and a fingerprint scanner to fuse face authentication and fingerprint authentication together [40]. Wilder et al. took facial thermogram from an inferred camera and face biometrics from a generic camera in authentication process [15,49]. Unlike the above approaches relying on the hardware sensors rarely deployed on mobile devices, our approach requires a front-facing camera, an accelerometer, and a gyroscope which are pervasively available on today's mobile devices.

The texture pattern based liveness detection techniques assume that the printed fake faces contain detectable texture patterns due to the printing process and the material printed on. Maatta et al. determined the liveness of a user based on the local binary patterns extracted from a single image [29]. IDIAP team took a facial video as input and the local binary patterns from each extracted frame in the video in order to build a global histogram for the video. The liveness of face is determined based on the global histogram [7]. The texture pattern based techniques usually require high-quality photos/videos captured in ideal lighting conditions, which may be hard to achieve in practice. In contrast, FaceLive takes the input videos in moderate quality which can be captured in varied lighting conditions.

The 3D face liveness indicator is based on the clue that a real face is a 3D object with depth characteristics. The detection of the characteristics about a 3D face is usually associated with optical flow analysis and changes of face views. A 3D face has the characteristic of optical flow that the motion speed of the central part of face is higher than the outer face region [24]. Along this line, Bao et al. proposed a liveness detection which analyzes the properties of the optical flow generated from a holistic 3D face [3]. Besides the holistic face, local facial landmarks are investigated in the optical flow analysis for liveness detection. Jee et al. proposed a liveness detection algorithm based on the shape variations of eye blinking, which is used in optical flow calculation [23]. Kollreider et al. proposed a liveness detection algorithm which analyzes the optical flow in detecting ears, nose, and mouth [26]. However, the optical flow analysis based approaches usually require high-quality input videos in ideal lighting conditions, which may be difficult to achieve in practice. Compared to these works, FaceLive takes input videos from a generic camera which can be easily achieved in practice.

On the other hand, the 3D characteristics about a real face can be detected in relative movements of the face. Findling et al. proposed to use multiple face views in face authentication. However, the proposed solution would still be vulnerable to the MFF-based attacks if an adversary obtains sufficient qualified face images with specific face views [12], while FaceLive is secure in such situation unless an adversary is able to obtain a qualified video and generate consistent device movement data. Chen et al. examined the 3D characteristics of nose in the liveness detection given the assumption that a real face has a 3D nose [8]. In order to determine the liveness of a user, the liveness detection mechanism compares the similarity between the direction changes of the mobile phone measured by the accelerometer, and the changes of a clear nose edge observed in the video from the camera. However, to produce a clear nose edge, a controlled lighting is required to cast the shadow of nose without any occlusion. This may be difficult to achieve in practice where the controlled lighting is not possible. The effectiveness of the liveness detection mechanism is also limited for those who have flat noses. Compared to this work, FaceLive can be used in complex lighting conditions and is robust to unsuccessful detection of partial facial landmarks including nose, eyes, and mouth because multiple facial landmarks are used in liveness detection.

The real-time response based approaches require interaction with users in real time. Pan et al. required users to blink their eyes in order to detect the liveness [36]. VeriFace, a popular face authentication software, asked users to rotate their heads in order to verify the liveness [27]. Unfortunately, these approaches are subject to the video-based attacks where adversaries may forge or replay facial videos containing the required interactions [38]. FaceLive can detect such video-based attacks effectively.

3. PRELIMINARIES

3.1 Face Authentication

As one of the most promising biometric-based user authentication, face authentication verifies a claimed identity of a user based on the facial features extracted from the images/videos about the user's face. A typical face authentication system usually consists of two subsystems including a face verification subsystem and a liveness detection subsystem, as shown in Figure 1.

The face verification subsystem takes the user's facial image/video as input through a camera and verifies it with the enrolled face biometrics for the claimed identity. This subsystem accepts the user if the input facial image/video matches the claimed identity and rejects the user otherwise. In the verification process, two key modules are involved, including the face detection module and the face matching module. The face detection module identifies the face region and removes the irrelevant parts of an image and then passes the processed image to the face matching module. The face matching module computes a similarity score for the input image by comparing the image with the enrolled face template and decides whether they belong to the same person. As face verification subsystem is designed to recognize the user from the input facial image/video and cannot detect forged biometrics, it is inherently vulnerable to the media based face forgery (MFF) where an adversary forges or replays a victim's facial images/video.

The liveness detection subsystem aims at preventing the MFF-based attacks. The liveness detection distinguishes between the faces of live users and the faces forged from facial images/videos.

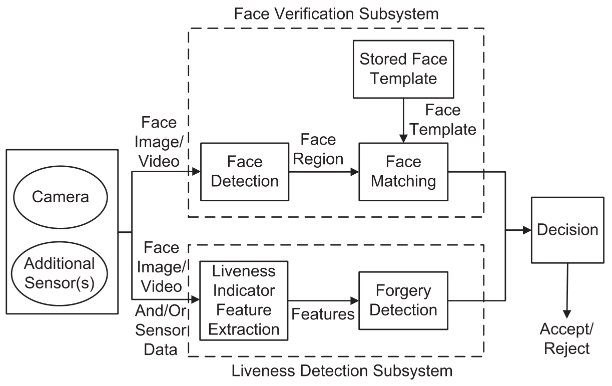


Figure 1: Work flow of a typical face authentication system

The liveness detection subsystem usually uses a camera or/and other additional sensors to capture the data about a live user during the face authentication. The captured data is analyzed by two key modules in this subsystem, including the liveness indicator feature extraction and the forgery detection. The liveness indicator feature extraction module takes the captured raw data as input and extracts feature information for liveness detection. Based on the feature information, the forgery detection module computes a liveness score and decides whether or not it is from a live user.

In the end, based on the decisions from the two subsystems, the face authentication system makes the final decision whether an authentication claim is accepted or rejected.

3.2 Media-based Facial Forgery and Threat Model

The media-based facial forgery (MFF) means that users' face biometrics can be forged by photos or videos containing the users' faces. The MFF poses a serious threat against face authentication systems as adversaries may forge or replay facial photos/videos to spoof face authentication.

Face authentication system has an intrinsic vulnerability against the MFF-base attacks during its face verification process. As shown in Figure 1, the face verification subsystem recognizes a user from an input facial photo/video. However, it cannot distinguish if the input facial photo/video is from a live user or from pre-recorded photo/video of the same user. Therefore, liveness detection is introduced into the face authentication system so as to thwart the MFF-based attacks, including photo-based attacks and video-based attacks.

The objective of liveness detection is to distinguish between the face biometrics freshly taken from a live user and the face biometrics forged from the user's facial photos/videos. Liveness detection is usually performed based on liveness indicators which can be derived from human physiological activities. There are four major types of liveness indicators, including multimodal, texture pattern, 3D face, and real-time response [24]. The multimodal usually requires a user to provide facial biometrics and additional biometric traits which are difficult to obtain by any adversary at the same time. The texture pattern can be used as a liveness indicator given the assumption that printed fake faces contain certain texture patterns which do not exist in real faces. The 3D face indicator is defined based on the fact that a real face is a 3D object with depth characteristics while a fake face in a photo/video is planar (2D). Finally, the real-time response can be used as a liveness indicator in the assumption that legitimate users can interact with the system in real time while it is difficult for fake faces to do so. In particular, eye blink and head rotation are two typical real-time response based liveness indicators which have been used in popular face authentication systems such as Google's FaceUnlock [16]. These liveness detection mechanisms require no additional hardware; they work with moderate image quality, and incur relatively low usability cost. They can effectively detect the photo-based attacks; however, they are still vulnerable to the video-based attacks.

The video-based attacks to face authentication systems pose significant security risks as massive amount of personal photos and videos are published online. It is likely that these photos and videos contain facial motions such as eye blink and head rotation which are natural physiology behaviors of humans. Even worse, the facial motions can be animated and synthesized through a single static facial photo from which a dynamic 3D face model is estimated [1,50]. Therefore, it is important for liveness detection to defend against the video-based attacks.

Our proposed liveness detection mechanism, FaceLive, aims to prevent both photo-based attacks and video-based attacks. FaceLive detects the attacks by measuring the consistency between the movements of mobile device and corresponding changes of head poses. The inertial sensor data of mobile device movements and the facial video of head pose changes are recorded simultaneously when a user moves a mobile device in front of his/her face.

In this paper, it is assumed that an adversary may obtain and replay a user's facial video containing continuous head pose changes during the video-based attacks. To perform the attacks, the adversary needs to hold and move a mobile device so as to generate the inertial sensor data for comparison with the facial video. According to the movements performed by the adversary, the video-based attacks are categorized as random-move attacks and imitation-move attacks. For the random-move attacks, an adversary randomly moves the mobile device to generate the inertial sensor data. For the imitation-move attacks, a more powerful adversary watches the facial video and attempts to move the phone accordingly in order to imitate the movements of the mobile device associated with the video. More details about these attacks are given in Section 5.1.

4. DESIGN OVERVIEW

FaceLive is a liveness detection mechanism that prevents the MFF-based attacks against face authentication systems on mobile devices based on measuring the consistency between a facial video about a user and the motion data about the user's mobile device. The video and the motion data are captured simultaneous and independently by a front-facing camera and the inertial sensors on the device, respectively. If a live user makes an authentication request, the changes of head poses estimated from the facial video should be consistent with the device movement estimated from the motion data.

FaceLive verifies the liveness of input facial biometrics based on the detection of a 3D face which is a liveness indicator commonly used by many liveness detection mechanisms as described in Section 3. To perform the liveness verification, a user needs to hold and move a mobile device over a short distance in front of his/her face. In this process, a front-facing camera on the device captures the video about the user's face from different camera angles. Meanwhile, the inertial sensors, including an accelerometer and a gyroscope, on the device record the motion data about the device movement concurrently. The changes of the user's head poses correlated with the device movement can be observed in the facial video if a real 3D face is in front of the device.

FaceLive consists of three modules, including Device Motion Estimator (DME), Head Pose Estimator (HPE), and Consistency

Analyzer (CA), as shown in Figure 2. The DME module takes the acceleration values from the accelerometer and the angular speed from the gyroscope as inputs and converts the input data into a motion vector representing the device movement over time. The HPE module extracts multiple frames from the facial video and estimates the head poses in these frames. The estimated angles of these head poses will be converted into a head pose vector over time. The CA module compares the motion vector and the head pose vector and extracts the features of correlation between the two vectors. Based on the extracted features, the CA module uses a classification algorithm to distinguish between a live user and an MFF-based attack. The detail of these modules is given in the rest of this section.

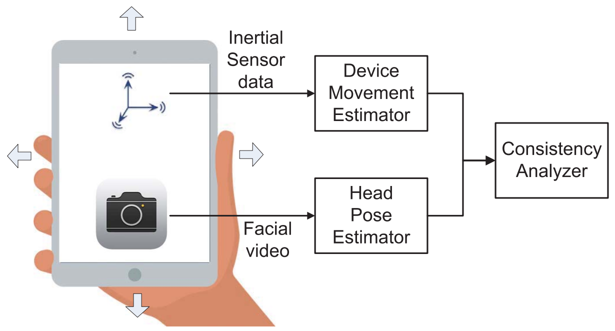


Figure 2: FaceLive mainly consist of device movement estimator, head pose estimation, and consistency analyzer

4.1 Device Motion Estimator

The Device Motion Estimator (DME) utilizes the motion data generated by an accelerometer and a gyroscope which are two inertial sensors widely available on mobile computing devices. Based on the motion data, DME estimates the movements of the device and outputs a motion vector for the device movement over time.

The motion data include acceleration values and angular speeds on three orthogonal axes (i.e. axis X, axis Y, and axis Z) in an inertial sensor coordinate system recorded by the accelerometer and the gyroscope, respectively. The typical inertial sensor coordinate system on the mobile device is defined relative to the screen of the device as shown in Figure 3. In the system, axis X and axis Y are in the same plane of the screen surface while axis Z is vertical to the screen surface. Axis X is horizontal and points to the right. Axis Y is vertical and points to the up. Axis Z points toward the outside of the front surface.

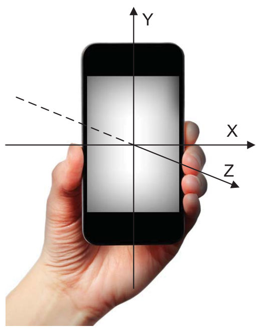


Figure 3: The coordinate system for a typical inertial sensor on a mobile phone

The acceleration values measured by the accelerometer may be affected by the gravity effect and the human physiological hand tremor. To reduce the effects of these factors, we preprocess the raw acceleration data as detailed below.

On the one hand, due to the gravity effect, an accelerometer on a mobile device reads the acceleration of $9.8m/s^2$ with a direction pointing to the earth center even when the device is stationary. To isolate and remove the contribution of gravity, we apply a low-pass filter and a high-pass filter [6]. More specifically, let $RA_{t_i} = (ra_{x,t_i}, ra_{y,t_i}, ra_{z,t_i})$ be the raw acceleration value at time t_i and $G_{t_i} = (g_{x,t_i}, g_{y,t_i}, g_{z,t_i})$ be the estimated gravity contribution at time t_i. Applying the low-pass filter, we have the estimation of the isolated gravity:

$$G_{t_i} = \alpha \cdot G_{t_{i-1}} + (1-\alpha) \cdot RA_{t_i} \tag{1}$$

where α is a smoothing factor. According to our experimental results, the filter performs best when $\alpha = 0.8$. Then, the estimated gravity is removed from the raw acceleration value by applying the high-pass filter:

$$FA_{t_i} = RA_{t_i} - G_{t_i} \tag{2}$$

where $FA_{t_i} = (fa_{x,t_i}, fa_{y,t_i}, fa_{z,t_i})$ is the filtered acceleration value.

On the other hand, when a user holds a mobile device, the slight involuntary movements of the device always occur as a result of physiological hand tremor by the user. The physiological hand tremor is usually caused by muscle contraction and relaxation, breath, arterial pulse, or the movements of other body parts [46]. Consequently, certain noises are introduced into the acceleration data. To remove the effects of the involuntary movements, we apply a threshold h to the filtered acceleration value FA_{t_i}. If an acceleration value $fa_{s,t_i} < h(fa_{s,t_i} \in FA_{t_i}, s \in \{x,y,z\})$ holds, it is considered as the result of involuntary movements and set to 0. Thus we have the preprocessed acceleration $A_{t_i} = (a_{x,t_i}, a_{y,t_i}, a_{z,t_i})$ with

$$a_{s,t_i} = \begin{cases} fa_{s,t_i} & fa_{s,t_i} \geq h \\ 0 & fa_{s,t_i} < h \end{cases} \tag{3}$$

where $s \in \{x,y,z\}$. In our experiments, the best estimation can be achieved when $h = 0.23m/s^2$.

Based on the preprocessed acceleration and the angular speed, we estimate the device movement using a dead-reckoning algorithm [25]. Given acceleration readings $a_{s,t_{i-1}}$ and a_{s,t_i} where $s \in \{x,y,z\}$, the movement on each axis during time t_{i-1} and t_i is calculated based the acceleration data by double integral. A trapezoidal rule is applied to approximate the calculation of the double integral in two stages [44]. In the first stage, the velocity v_{s,t_i} on axis s at time t_i is obtained by

$$v_{s,t_i} = v_{s,t_{i-1}} \cdot \cos\varphi_{s,t_i} + (a_{s,t_{i-1}} + a_{s,t_i}) \cdot (t_i - t_{i-1})/2 \tag{4}$$

where $v_{s,t_{i-1}}$ is the velocity at time t_{i-1}, and φ_{s,t_i} is the angle between axis s of the coordinate system L in the interval (t_{i-2}, t_{i-1}) and axis s of the coordinate system L' in the interval (t_{i-1}, t_i). Note that $\cos\varphi_{s,t_i}$ can be calculated based on the angular speed recorded by the gyroscope. The initial velocity speed v_{s,t_0} is assumed to be 0 because the device movement has not begun yet at time t_0. In the second stage, the movement distance $d_{s,t_{i-1}}$ on axis s at time t_i can be calculated as

$$d_{s,t_i} = (v_{s,t_{i-1}} \cdot \cos\varphi_{s,t_i} + v_{s,t_i}) \cdot (t_i - t_{i-1})/2 \tag{5}$$

Let $R_{t_i} = (r_{x,t_i}, r_{y,t_i}, r_{z,t_i})$ be the angular speed generated by the gyroscope. The rotation angle along each axis can be calculated by the trapezoidal rule for integral approximation as follows

$$\theta_{s,t_i} = (r_{s,t_{i-1}} + r_{s,t_i}) \cdot (t_i - t_{i-1})/2 \tag{6}$$

where $s \in \{x, y, z\}$. Note $(\theta_{x,t_i}, \theta_{y,t_i}, \theta_{z,t_i})$ is also called as Cardan angles which are typical for rotation of a 3D coordinate system [47]. Then, we have

$$\begin{aligned} \cos\varphi_{x,t_i} &= \cos\theta_{y,t_i} \cdot \cos\theta_{z,t_i} \\ \cos\varphi_{y,t_i} &= \cos\theta_{x,t_i} \cdot \cos\theta_{z,t_i} \\ \cos\varphi_{z,t_i} &= \cos\theta_{x,t_i} \cdot \cos\theta_{y,t_i} \end{aligned} \tag{7}$$

We can estimate the movement distance $d_{s,t_{i-1}}$ by combining Equation (5) and Equation (7). In this paper, we exclude the movements along axis Z because such movements do not change the views of a 3D face but only scale up/down the size of a single facial view in the photo. Finally, DME outputs a device motion vector (D_x, D_y) between t_0 and t_m for axis X and axis Y, where $D_s = (d_{s,t_1}, d_{s,t_2}, ..., d_{s,t_m})$ and $s = \{x, y\}$.

4.2 Head Pose Estimator

As a mobile device moves in front of a user's face, the camera on the device captures multiple video frames about the user's face at different camera angles. Due to the 3D characteristics of human face, different views of the face (i.e. head poses) can be observed in the video frames. The Head Pose Estimator (HPE) analyzes the frames from the facial video and estimates head poses in these frames. The changes of these head poses are supposed to be correlated with the device movements. HPE calculates a head pose vector representing the changes of the head poses over time.

In order to estimate the head poses in the facial video frames, HPE takes two steps: facial landmark localization and head pose estimation. In the first step, a set of facial landmarks is located in each frame, including inner/outer corners of left/right eyes, nose, left/right corners of lips, and lower facial contours, as illustrated in Figure 4. We use FaceTracker, an open-source library, to automatically extract the location of these facial landmarks in real-time. FaceTracker is an implementation of the constrained local model (CLM) which utilizes the subspace constrained mean-shifts algorithm to optimize the CLM fitting and minimize the misalignment error over all facial landmarks [41]. The location of these facial landmarks is fed into the head pose estimation algorithm in the next step. In the second step, we estimate the head pose in a frame using the 2D head pose estimation algorithm [35, 39]. This head pose estimation algorithm has advantages of low theoretical mean error of $2.9°$, real-time automatic estimation process, and identity independent feature. The head pose estimation algorithm analyzes the projection from points in a 3D coordinate system (i.e. camera) into a 2D coordinate system (i.e. image) using a perspective transformation [35, 39]. Given a set of points from a face in the 3D coordinate system and the projection of these points in the 2D coordinate system, the transformation between the two systems can be estimated with the following equations:

$$\tilde{m} = A[\mathsf{R}|\mathsf{T}]\tilde{M} \tag{8}$$

where $\tilde{m} = (u\ v\ 1)^T$ is a 2D point (u, v) in homogenous coordinates, $A = \begin{pmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{pmatrix}$ is an internal calibration matrix, $R = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix}$ is a rotation matrix, $T = (t_1\ t_2\ t_3)^T$ is a translation vector, and $\tilde{M} = (x\ y\ z\ 1)^T$ is a 3D point (x, y, z) in homogenous coordinates. A can be obtained from the camera settings where f in A is the focal length of the camera in pixel unit while (c_x, c_y) is the center point of the 2D image in pixels. The 3D point coordinates are obtained from a pre-acquired 3D face model.

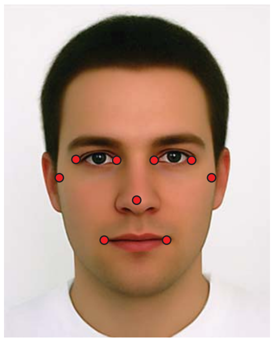


Figure 4: Facial landmarks used by the head pose estimation algorithm

Therefore, given a set of locations of the facial landmarks in a 2D facial video frame and corresponding 3D coordinates in the 3D face model, the head pose estimation algorithm can estimate the rotation matrix R and the translation vector T. Based on the estimation of R and T, the 3 degrees of freedom (3DoF) of the head pose in the frame can be calculated [35]. The 3DoF of a head pose represents the rotation of the head relative to the front head pose from three rotation angles, including Yaw, $Pitch$, and $Roll$, as shown in Figure 5. We use Yaw and $Pitch$ which are caused by horizontal movements (along axis X) and vertical movements (along axis Y) of mobile device, respectively. We do not use $Roll$ because it does not contribute to the 3D face detection. A head pose change vector is derived from the input facial video of $n + 1$ frames labeled between 0 and n denoted as (H_{yaw}, H_{pitch}), where $H_w = (\beta_{w,1}, \beta_{w,2}, ..., \beta_{w,n})$ and $\beta_{w,i}$ is the difference between the head pose in frame $i - 1$ and in frame i for $w = \{yaw, pitch\}$.

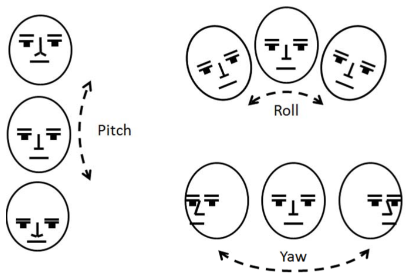


Figure 5: Head rotation in 3DoF [33]

4.3 Consistency Analyzer

Based on the device motion vector (D_x, D_y) from DME and the head pose change vector (H_{yaw}, H_{pitch}) from HPE, the Consistency Analyzer (CA) examines the correlation between the two vectors and uses a classifier to make the final decision based on the correlation.

4.3.1 Correlation Analysis

When a user holds and moves a mobile device in front of a real face over a distance, the changes of the views of the face can be observed in the facial video. In particular, the horizontal movement of the device (along axis X of the device) leads to changes in yaw while the vertical movement of the device (along axis Y of the device) leads to changes in pitch. The distance d of the device movements can be correlated to the angle difference γ of the head poses in the following geometrical relation

$$d = \gamma \cdot D_{FD} \tag{9}$$

where D_{FD} is the distance between the camera on the device and the user's face. The value of D_{FD} is difficult to estimate accurately from the facial video due to the varied size of human faces. We thus introduce a synthetic camera based technique to convert the head pose change vector into a synthetic camera motion vector which is compared to the device motion vector. Consequently, the correlation analysis takes three steps, including synthetic camera motion vector conversion, data alignment, and correlation feature extraction.

In the first step, we convert the head pose change vector into a motion vector by a synthetic camera. A synthetic camera is an imaginary camera which moves and captures facial video synchronously with the mobile device at all time. The distance between the synthetic camera and the face is assumed to be $D^{'}_{FD}$ which could be different from the real distance D_{FD}. More specifically, in each time interval (t_{i-1}, t_i), the synthetic camera movements always result in the same head pose changes as the real device movements. The track of the device movements can be approximated by an arc with the circle center located at the center the 3D face. Since the synthetic camera moves synchronously with the real device, the movements of the synthetic camera $D^{'}_s$ along the horizontal axis or the vertical axis can be calculated based on the head pose change vector $H_w = (\beta_{w,1}, \beta_{w,2}, ..., \beta_{w,n})$ and Equation (9). Given a fixed $D^{'}_{FD}$, the synthetic camera motion vector is $D^{'}_s = (\beta_{w,1} \cdot D^{'}_{FD}, \beta_{w,2} \cdot D^{'}_{FD}, ..., \beta_{w,n} \cdot D^{'}_{FD})$, where $s = x$ if $w = yaw$ and $s = y$ if $w = pitch$.

The second step is data alignment. Because the inertial sensors and the front-facing camera on a mobile device usually work at different sampling rate, the device motion vector $D_s = (d_{s,t_1}, d_{s,t_2}, ..., d_{s,t_m})$ estimated from the sensor data should be aligned with $D'_s = (d'_{s,t_1}, d'_{s,t_2}, ..., d'_{s,t_m})$ where $d'_{s,t_i} = \beta_{w,i} \cdot D^{'}_{FD}$,$s = \{x, y\}$. A dynamic time warping (DTW) algorithm is used for the alignment. The DTW algorithm is a well-known technique to find an optimal alignment minimizing the warping distance between two time-dependent sequences which may vary in time or magnitude [32]. Given D_s and D'_s, the objective function of the DTW algorithm minimizes the warping distance:

$$DTW(D_s, D'_s) = \min\{c_p(D_s, D'_s) = \sum_{l=1}^{L} c(d_{s,t_l}, d'_{s,t_l})\} \tag{10}$$

where p is an (m, n)-warping path. The optimal warping path for the alignment and the minimum warping distance $c^*_p(D_s, D'_s)$ are returned by the DTW algorithm.

The third step is correlation feature extraction. Given the two aligned motion vectors D_s and D'_s, the ratio r_{t_i} between the two vectors in each time interval (t_{i-1}, t_i) is

$$r_{t_i} = d'_{t_i}/d_{t_i} = D'_{FD}/D_{FD} \tag{11}$$

for all $d'_{t_i} \in D'_s$ and $d_{t_i} \in D_s$. Because D'_{FD} and D_{FD} are two constants, we have $r_{t_i} = r_{t_j}$ for all $r_{t_i}, r_{t_j} \in \{r_{t_1}, r_{t_2}, ..., r_{t_N}\}$. From the ratio vector $(r_{t_1}, r_{t_2}, ..., r_{t_N})$, we can calculate the mean value $r_{mean} = 1/N \cdot \sum_{i=1}^{N} r_{t_i}$, maximum value $r_{max} = \max\{r_{t_1}, r_{t_2}, ..., r_{t_N}\}$, minimum value $r_{min} = \min\{r_{t_1}, r_{t_2}, ..., r_{t_N}\}$, and standard deviation $r_{sd} = \sqrt{1/N \cdot \sum_{i=1}^{N}(r_{t_i} - r_{mean})^2}$.

4.3.2 *Classification*

In the correlation analysis, a set of feature parameters are generated for measuring the correlation between the device motion vector and the head pose change vector. The feature parameters are shown in Table 1. In particular, the DTW warping distance c^*_p measures the similarity between the device motion vector and the head pose change vector which may vary in magnitude or time [32]. The mean ratio r_{mean}, maximum ratio r_{max}, minimum ratio r_{min}, and standard variation of ratio r_{sd} are basic descriptive statistics to summarize the ratio values between the two input vectors in the correlation analysis [30]. The cumulative shift of device d_x (or d_y) and the cumulative shift of synthetic camera d'_x (or d'_y) describe the total movement distances of the device and the synthetic camera. All these feature parameters are fed into a classification algorithm so as to determine whether the input motion data matches the input facial video for the liveness verification.

Table 1: Feature parameters

Parameter	Notation
DTW warping distance	c^*_p
Mean ratio	r_{mean}
Maximum ratio	r_{max}
Minimum ratio	r_{min}
Standard variation of ratio	r_{sd}
Cumulative shift of device on X (or Y) axis	d_x (or d_y)
Cumulative shift of synthetic camera on X (or Y) axis	d'_x (or d'_y)

In FaceLive, the classification algorithm can be chosen from Bayesian Network, Binomial Logistic Regression, and Multilayer Perceptron. In particular, Bayesian network (BN) is a probabilistic graphical model of joint multivariate probability distributions [13]. BN is capable of handling incomplete data set and easily recognizing direct dependencies. Binomial logistic regression (BLR) is a classical probabilistic statistical classification model which predicts the outcome for a dependent variable based on multiple predictor variables [21]. BLR can be used as a parametric analytic tool to assess the significance of an individual predictor variable and to provide probability outcomes. Multilayer perceptron (MLP) is a feedforward neural network model which maps a set of input data onto a set of outputs [14]. Finally, MLP can be used to process complicated and imprecise data in an adaptive learning.

5. DATA COLLECTION AND EVALUATION

An IRB-approved user study is conducted to evaluate the performance of FaceLive in terms of practicality and robustness.

5.1 Data Collection

Our user study involves 73 participants, including 42 males and 31 females whose ages range between 19 and 36. The participants are recruited using recruiting emails. Each participant spends around 60 minutes in a quiet room in the study and is paid with 10 dollars as compensation. There are four parts in the study. After the completion of each part, the participants have a short break of 1-3 minutes before moving to the next part. The detail of the user study is given below.

In the first part, we capture the participants' facial photos with 35 controlled head poses by using a Canon PowerShot A2600 (16.0-megapixels camera). The resulting photos are 4608×3456 pixels in size with inner pupil distance of the subjects typically exceeding 400 pixels. The 35 head poses are specified by both *yaw* (horizontal) and *pitch* (vertical) rotation and represented as $(\beta_{yaw}, \beta_{pitch})$. The value range of β_{yaw} includes 0°, 10° to left/right, 20° to left/right, and 30° to left/right while the value range of β_{pitch} includes 0°, 10° to up/down, and 20° to up/down. These rotation boundary values are chosen according to the common restriction of the face authentication systems [1] where a participant should not pass the face authentication if β_{yaw} exceeds 30° or β_{pitch} exceeds 20°.

A continuous lighting system is used to eliminate the shadow on each participant's face. In order to control the head rotation of a participant, we use a helmet equipped with a gyroscope. The use of gyroscope has significant advantages over other head rotation control methods, including accuracy of less than 1° and robust to metallic interference [33]. For each head pose, the participants are asked to face to the Canon camera and adjust their heads to frontal position in the way similar to [18]. Next, the participants rotate their heads to the required angles with help of the gyroscope on their helmets. The gyroscope generates real-time rotation angles and broadcast the angle values via WiFi. The rotation angles are received and displayed on an iPad screen and shown to the participants. Then, the participants are asked to hold their head poses, while we help remove their helmets gently and quickly so as to avoid any significant movements of the heads during the helmet removal. At last, the facial photos of each head pose are captured immediately. In order to avoid head movements when capturing head pose images, we capture an image before removing helmet and another image after. If any head movement is observed by comparing the images, we recapture the images until no head movement is observed. The facial photos will be used as the ground truth for evaluating the accuracy of the head pose estimation algorithm in HPE of FaceLive.

In the second part, we collect the facial videos and inertial sensor data from the participants' trials of FaceLive. Each participant is asked to perform 36 trials of FaceLive with 4 controlled device movement distances and 3 controlled device positions using an HTC One X smartphone, which is equipped with a 1.3-megapixel front-facing camera (720p for video), 4.7-inch screen, 1.5 GHz quad-core CPU, and Android 4.4.2 operating system. For each trial of FaceLive, a participant holds and moves the smartphone horizontally (along axis X of the smartphone) over a given movement distance D_{MD} while the distance between the smartphone and his/her face is set to D_{FD}. We do not consider the movements in vertical direction (along axis Y of the smartphone) and diagonal directions because more than half of the participants have difficulties in performing these movements based on our pilot study.

The range of D_{MD} includes 10cm, 20cm, 30cm, and 40cm while the range of D_{FD} includes 30cm, 40cm, and 50cm. These values are chosen based on the common behaviors of participants. According to our pilot study, more than 70% of participants have difficulties in moving the smartphone for $D_{MD} > 40cm$ or $D_{FD} > 50cm$ due to the limitation of the length of their arms. Since the FaceLive relies on the movement of the smartphone, D_{MD} cannot be 0. Before each trial, a researcher demonstrates the required movement in front of a participant, and the participant is required to practice the movement. It is discovered that 60% of participants' faces cannot be fully captured by the camera if $D_{FD} < 30cm$. During the movement, the participant's facial video which is captured by the front-facing camera is displayed on the screen in real time and viewed by the participant so that the participant can adjust the smartphone and ensure his/her face is always fully captured by the front-facing camera. In order to control the moving distance, we mark the required distance along the horizontal axis on the wall and the required distance between the participant's head and the smartphone on the floor. We ensure no significant head rotation is involved when a participant moves the smartphone in a way similar to [18]. Given D_{MD} and D_{FD}, a participant is required to perform 3 trials of FaceLive. Both facial video and inertial sensor data are recorded simultaneously during each trial. In total, we collect data from 2628 trials of legitimate authentication.

In the third part, we record the inertial sensor data about the MFF-based attacks performed by participants. In particular, the participants are asked to perform the random-move attacks first and then the imitation-move attacks. In the random-move attacks, each participant moves the smartphone horizontally in random speed and direction (left/right) for three times as guesses. In the imitation-move attacks, the participant watches a facial video of another participant on the smartphone and tries to move the smartphone in the direction and speed as observed based on the head pose changes in the video in order to imitate the movements associated with the video. The participant is required to perform 3 trials for each given video. The inertial sensor data is automatically recorded in the smartphone during the participant's trials. We collect data from 5256 trials in total, including 2628 trials from the random-move attacks and 2628 trials from the imitation-move attacks.

In the final part, each participant is asked to fill in a questionnaire with a 5-point Likert scale. The user study in this part is to collect users' perception on the usage of FaceLive.

5.2 Experimental Results

Based on the data collected from FaceLive trials and MFF-based attacks, we evaluate the practicality and robustness of FaceLive, where Bayesian Network (BN), Binomial Logistic Regression (BLR), and Multilayer Perceptron (MLP) are used as classifiers to distinguish between legitimate authentication trials and MFF-based attacks. Each attack trial is assigned to the *positive* class while each legitimate trial is assigned to the *negative* class.

In terms of practicality, we show that FaceLive can effectively detect the MFF-based attacks without requiring any additional hardware. The evaluation is based on the attack datasets, including the random-move dataset and the imitation-move dataset as described in Section 5.1. The evaluation metrics include *Receiver Operating Characteristic* (ROC) curve and *Equal Error Rate* (EER). In particular, the ROC curve illustrates the relation between *True Positive Rate* (TPR) and *False Alarm Rate* (FAR) at various threshold settings, where TPR measures the proportion of correctly identified positives and FAR measures the proportion of negatives which are falsely assigned to the positive class [19]. EER is the rate at which false positive rate and false negative rate are equal [5,22]. The value of EER can be derived from a ROC curve. The EER of FaceLive is as low as 4.7% (when BN is used as the classifier). FaceLive achieves TPR of 97.7% while its corresponding FAR is 6.1%.

To explain why EER cannot reach 0%, we evaluate the accuracy of device movement estimation and head pose estimation, which are two critical components of FaceLive. We discover that the mean error of the device movement estimation is at most 23% and the mean error of the head pose estimation is 7.1° for horizontal rotation, and 7.9° for vertical rotation. These values will be improved in the future when more accurate inertial sensors are deployed on mobile devices. Meanwhile the current values are reasonable for FaceLive to detection the MFF-based attacks effectively.

The practicality of FaceLive is also demonstrated with satisfactory usability results. The average time spent on device movements is 3.3 seconds. The Likert scores for the controlled setting of the device movements are as high as 4.8 and 4.3 out of 5 in the preferred settings.

We also evaluate the robustness of FaceLive with respect to unsuccessful detection of the partial facial landmarks and head instability. These may happen frequently in real-world settings and have a negative impact on the accuracy of FaceLive. Our results show that the EER of FaceLive remains in the range of [7.2%,7.5%] when a single facial landmark (mouth, nose, or eyes) is missing in liveness detection. The EER of FaceLive is below 10% when the accumulative error caused by head instability is below 16°.

5.2.1 Detecting MFF-based Attacks

Using an accelerometer, a gyroscope, and a front-facing camera, FaceLive measures the consistency between device movements and head pose changes in order to authenticate a live user's face. FaceLive aims to distinguish between legitimate authentication and MFF-based attacks, including photo-based attacks and video-based attacks. In the photo-base attacks, an adversary replays a single photo of the legitimate user. Because FaceLive requires changing head poses in a facial video for liveness verification, it is straightforward that FaceLive can prevent the photo-based attacks as single photo cannot produce multiple head poses. In the following, we focus on detecting the video-based attacks where a more powerful adversary replays a pre-recorded facial video containing the required head pose changes.

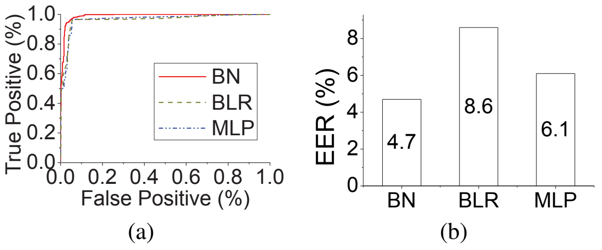


Figure 6: ROC curves and EER of FaceLive using different classifiers

Figure 6(a) shows the ROC curves of FaceLive using Bayesian network (BN), BLR (Binomial logistic regression), and MLP (Multilayer perceptron) as the classifiers separately. Figure 6(b) shows that FaceLive with BN outperforms FaceLive with BLR and MLP. The EER of FaceLive with BN is 4.7%, which is lower than BLR (8.6%) and MLP (6.1%). In order to examine the significance of the feature parameters used by the classifier, we further run BLR on SAS software. The likelihood ratio test results and the wald statistics [21] for FaceLive with BLR classifier are all smaller than 0.0001.

Our statistical analysis shows that the most influential parameters are the warping distance c_p^* (p-value $p < 0.0001$), the mean ratio value r_{mean} ($p = 0.001$), the maximum ratio value r_{max} ($p = 0.001$), the minimum ratio value r_{min} ($p = 0.001$), and the standard deviation of the ratio vector r_{sd} ($p = 0.014$). The results of the parameter significance test are given in Table 2.

The warping distance c_p^*, the mean ratio value r_{mean}, the maximum ratio value r_{max}, and the standard deviation r_{sd} have a positive impact on the probability of determining a trial as an attack while the minimum ratio value r_{min} has a negative impact on the same probability. The similarity between the device movements and the head pose changes decreases as the values of the positive parameters increase and as the value of the negative parameter decreases. The attack trials generally have a lower similarity than the legitimate trials. The trials with lower similarity are more likely to be classified as attacks and rejected by FaceLive.

Table 2: The results of statistical tests for the feature parameters. The statistically significant results are marked with ★

Parameter	Coefficient	P value
c_p^*	0.3242	< 0.0001★
r_{mean}	0.0034	0.001★
r_{max}	0.000847	0.001★
r_{min}	-0.000848	0.001★
r_{sd}	5.69E-14	0.014★
d_x	-0.4895	0.1128
d'_x	-0.8972	0.8075

We further evaluate the detection of the random-move attacks and the imitation-move attacks, which are two types of the video-based attacks. Our results reveal that FaceLive detects the random-move attacks more effectively than the imitation-move attacks, as shown in Figure 7. For the random-move attacks, the EER of FaceLive is as low as 1%. For the imitation-move attacks, the lowest EER of FaceLive is 6.1%.

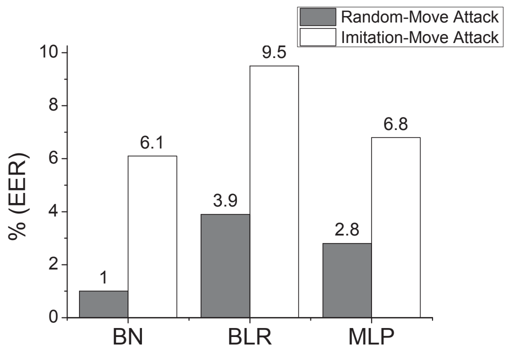
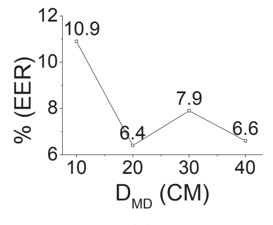


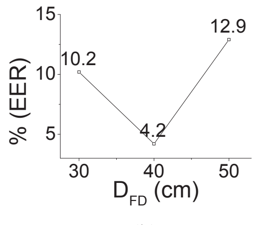
Figure 7: EER of FaceLive under different attacks

Users' behaviors, including the distance D_{MD} of the device movements and the distance D_{FD} between the device and the users' faces, may affect the detection of the MFF-based attacks, especially the accuracy of the device movement estimation and the head pose estimation. As shown in Figure 8(a), the EER of FaceLive is relatively low when D_{MD} is equal to 20cm, 30cm, 40cm as compared to D_{MD} = 10cm. The average EER rates for D_{MD} = 20cm, 30cm, 40cm are 6.4%, 7.9%, and 6.6%, respectively. When D_{MD}=10cm, the average EER rate increases to 10.9%. Recall that FaceLive detects the attacks by analyzing the correlation between the device movements and the head pose changes. If the distance of the device movements is too short, the head pose changes observed in the video are insufficient, which leads to relatively high EER.

The distance D_{FD} between the device and the users' faces is another user behavior factor which affects FaceLive. As shown in Figure 8(b), FaceLive achieves the lowest EER of 4.2% when D_{FD}=40cm. The EER values increase to 10.2% and 12.9% when D_{FD} is equal to 30cm and 50cm, respectively. The distance D_{FD} may affect the quality of facial videos and thus the accuracy of head pose estimation in HPE. As the device is drawn closer to a user's face, the face region occupies more pixels in a video frame, and the face region is easier to go out of the frame even with minor rotation of the device. The incomplete facial region in the video frame may reduce the accuracy of the head pose estimation algorithms [33].



(a)



(b)

Figure 8: EER of FaceLive under controlled movements

On the other hand, if a user keeps the device farther away from the user's face, the face region occupies fewer pixels in a frame. This may lead to lower accuracy in detecting important facial landmarks which are used by the head pose estimation algorithms [33].

In order to analyze the quality of the facial videos captured at different distance D_{FD}, we examine the bad frame rate of the videos. The bad frame rate is the proportion of the facial frames from which the head poses cannot be estimated by the head pose estimation algorithm in HPE. Figure 9 shows that the bad frame rate is the lowest at D_{FD} =40cm, which is 3.7%. In contrast, the bad frame rates at D_{FD} =30cm, 50cm are higher, which are 6.6% and 12.2%, respectively. The higher the bad frame rate, the higher the EER of FaceLive.

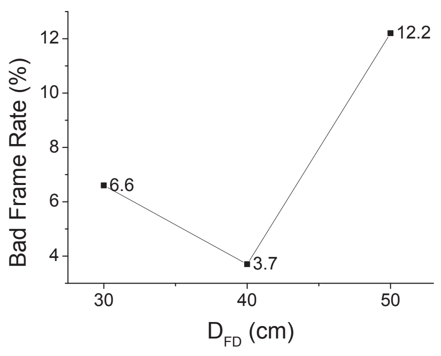


Figure 9: Bad frame rate at different D_{FD}

5.2.2 Accuracy of Device Movement Estimation and Head Pose Estimation

FaceLive cannot detect the MFF-based attacks with 100% accuracy. This is mainly due to the errors in device movement estimation and head pose estimation.

The device movements are estimated based on the acceleration measured by an accelerometer and the angular speed measured by a gyroscope on the mobile device. We evaluate the accuracy of the device movement estimation of FaceLive in a follow-up experiment. In this experiment, a smartphone is held in portrait orientation where axis X of the smartphone is set along the horizontal direction while axis Y of the smartphone is set along the gravity direction. The smartphone moves horizontally over 10cm, 20cm, 30cm, and 40cm, respectively which are marked on the wall and used in our study. During the movements, the acceleration values and the angular speed data of the smartphone are recorded by the inertial sensors. The cumulative shift of the movements in axis X is estimated by the device movement estimation algorithm. On average, the device movement estimation algorithm achieves a mean error of 4.8cm. As the distance of the movements increases, the mean error of the algorithm increases as well, which is shown in Figure 10. The device movement estimation is based on the dead-reckoning algorithm which estimates the current position based a previously determined position. The dead-reckoning algorithm is subject to cumulative errors which increase over time [25]. As the device moves over a longer distance, the cumulative errors increase, too.

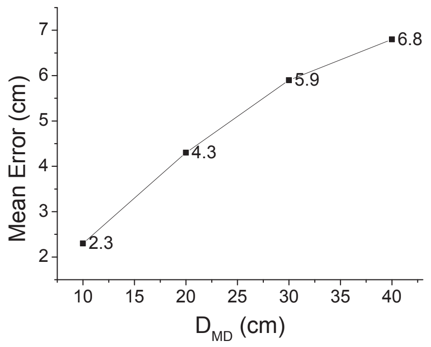


Figure 10: Accuracy of the device movement estimation

The head pose changes are estimated from a given facial video using the head pose estimation algorithm in HPE. In order to evaluate the accuracy of the head pose estimation algorithm, we use the facial photos with controlled head poses collected in our user study. Our results show that the head pose estimation achieves a mean error of 7.1° in *yaw* (horizontal rotation) and 7.9° in *pitch* (vertical rotation).

The errors in device movement estimation and head pose estimation lead to the errors of FaceLive in detecting the MFF-based attacks. Our results show that these errors are reasonably low, and the results could be further improved once high accurate inertial sensors become available on mobile devices.

5.2.3 Usability

We measure the usability of FaceLive using three metrics, including the average time of the device movement required for the liveness verification, the average Likert scores for the device movement distance, and the average Likert scores for the relative distance between device and face.

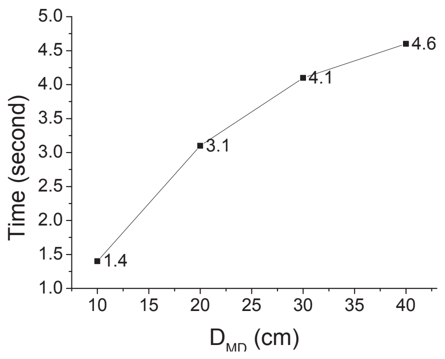
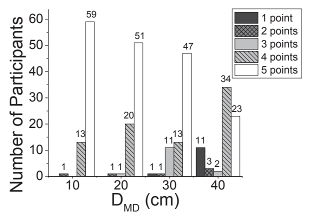
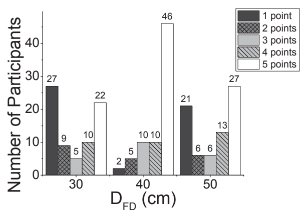


Figure 11: The time of movement at different D_{MD}



(a)



(b)

Figure 12: 5-point Likert scale for the controlled movements

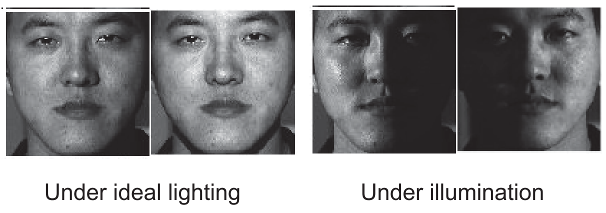


Figure 13: Face photos in ideal lighting condition and under illumination in varied environments [18]

The liveness detection process spends most of its time on recording the required device movements and facial video. Thus the time spent on device movements affects users' experience. According to our results, the time of device movements for FaceLive is 3.3 seconds on average. As the distance of the movement increases, the time spent increases as well, which is shown in Figure 11. In order to evaluate the users' comfortability with the distances they move over in the study, the users are asked to assign 5-point Likert scores to each movement distance in the question "how comfortable for you to move the smartphone over the distance performed in the study." The Likert scores are numbers varying from 1 to 5. The higher the Likert scores, the higher the users' comfortability. As shown in Figure 12(a), the users prefer a shorter distance for the device movements in general.

Another factor is the relative distance between device and face. We evaluate the users' comfortability with this relative distance using a question "how comfortable for you to move the smartphone with the relative distance between the smartphone and your face performed in the study." Our results show that the users prefer 40cm between the device and their faces, which is shown in Figure 12(b). The distance of 30cm is least preferable. At this distance, users may need to move and adjust their devices carefully because the face region could easily move out of a video frame with minor rotation of the devices.

5.2.4 Robustness

It is important for liveness detection on mobile devices to be robust to complex lighting conditions and head instability in varied environments. The complex lighting conditions may introduce illumination as shown in Figure 13, and further lead to low accuracy in detecting important facial landmarks, such as eyes, nose, and mouth. This will affect the accuracy of facial recognition and head pose estimation [33, 50]. On the other hand, the head instability may introduce errors into head pose estimation. In this section, we evaluate the robustness of FaceLive in two situations, including the unsuccessful detection of important facial landmarks and the head rotation when moving the device. The facial landmarks include eyes, nose, and mouth which are important features used by many head pose estimation algorithms [33]. In order to simulate the first situation, we intentionally remove from the collected datasets the localization information about eyes, nose, mouth, and the combinations of every two types of the facial landmarks. In order to simulate the second situation, we intentionally add a uniformly perturbed error to each head pose measured in FaceLive.

Figure 14 show that FaceLive achieves the EER rate in the range of [7.2%,7.5%] when the detection of a single type of facial landmarks is unsuccessful and the coordinate information of the facial landmarks is missing.

In particular, the missing localization of nose has a slightly more significant impact on FaceLive than the missing localization of eyes and mouth. The reason is that during head rotation, significant movements of nose can be observed. Nose is the part which sticks out most from the face surface and is usually closest to a camera capturing the face. As the head is rotating, the velocity of the nose is higher than the other parts on the face according to the geometric relationship between linear and angular motions [4]. On the other hand, from the view of the camera, the nose can be observed faster than the other parts on the face during the head rotation according to the theory of optical flow which describes a visual phenomenon that the objects closer to an observer (e.g. eyes, camera) appear to move faster than other distant objects [20].

The unsuccessful detection of eyes and mouth also decreases the accuracy of head pose estimation. Accurate localization of eyes is important for the alignment process in most head pose estimation algorithms [1]. On the other hand, mouth is a trackable facial landmark for head pose estimation because it exhibits a contrasting region and is usually correlated with the orientation of the head [33].

As the localization information about more facial landmarks is missing, the accuracy of head pose estimation decreases, which has a negative impact on FaceLive. As it is shown in Figure 14, EER of FaceLive increases to at least 18.1% when the detection of any two types of the facial landmarks is unsuccessful. If the localization information about all of the facial landmarks is missing, the head poses cannot be correctly estimated by any head pose estimation algorithms, which leads to the failure of FaceLive.

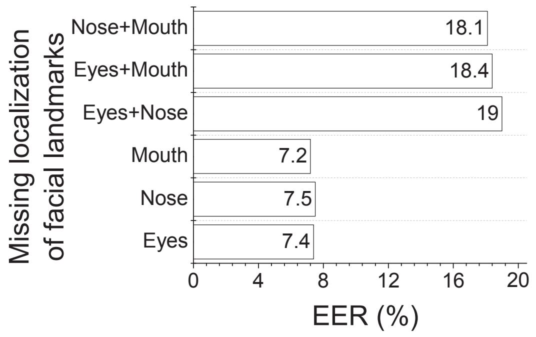


Figure 14: EER of FaceLive under missing localization of the facial landmarks

On the other hand, FaceLive is robust to the instability of head movements. To simulate the instability of head movements, we add uniformly perturbed error to each head pose in the head pose change vector calculated by HPE. Figure 15 shows that FaceLive achieves EER of no more than 9.6% when the total error of the head instability is no more than 12°. The EER of FaceLive increases to 12.5% when the total error reaches to 20°.

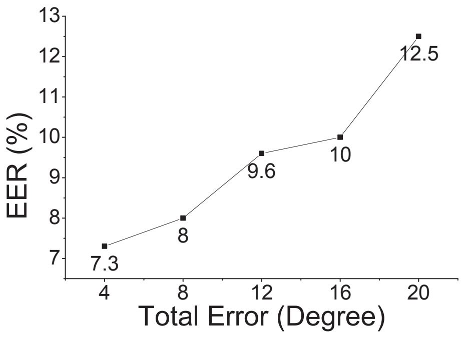


Figure 15: EER of FaceLive under head instability

6. DISCUSSION ON LIMITATIONS

Ecological validity is a challenge to any user study, like most prior research [2, 17, 45], our study mainly recruits students in university. These participants are usually more active in using mobile devices. Thus the performance evaluation may vary with other populations.

In our user study, the facial photos with controlled head poses are used the ground truth for evaluating the accuracy of the head pose estimation algorithm. The collection of the facial photos with precisely controlled head poses is still challenging [33]. Like the prior head pose data sets [18, 43], the accuracy of the head poses in our data set may be affected by the unconscious movement of human beings and imperfect capability of the participants to accurately direct their heads. Due to the involuntary hand tremors of human being and users' imperfect capability of controlling hand movements [8,37], it is also not easy to collect facial videos and device movement data with precisely controlled distance for device movements and precisely controlled distance between the device and users' faces.

FaceLive requires motion data and facial video so as to analyze the 3D face indicator of a user. The dominant factor in FaceLive performance is the time for performing the movement and processing the facial video, which is similar to most face liveness detection methods based on 3D face indicator [3, 8, 23, 24]. In our experiments, the movement in each trial takes 3.3s on average while the detection of face and the estimation of head pose can be performed in parallel when the facial video is recorded. The authentication time can be reduced by processing sampled video frames instead of processing all video frames. FaceLive can work with any face verification subsystem in face authentication. Note that the time taken by the face verification subsystem for the recognition of multiple face images is negligible compared to the time taken for performing the movement in FaceLive. For example, the computation time of PCA-based face recognition is 0.7ms for processing 100 input face images [9]. As the performance and usability of face verification further improve, it is easy to combine it with FaceLive for better performance in face authentication [10, 31].

Regarding the liveness detection errors, it is desired to control noise from device movement estimation and head pose estimation. The accuracy of device movement estimation is mainly affected by the cumulative errors in the dead-reckoning based estimation algorithms and the accuracy of the inertial sensors [25]. All dead-reckoning based algorithms are subject to cumulative errors because they estimate the current position based on a pre-determined position [25]. The errors can increase rapidly as time elapses. The accuracy of the inertial sensors used on existing mobile devices is still limited [42]. The impact of gravity is not well handled by these inertial sensors. This can significantly affect the accuracy of the device movement estimation, especially the movement estimation in the direction of gravity. Fortunately, the accuracy of the inertial sensors on mobile devices is improving as more applications, such as electronic games, require the support of inertial sensors with relatively high accuracy; meanwhile, the cost of these sensors keeps decreasing. This trend will lead to the improved accuracy of the device movement estimation algorithm in FaceLive. On the other hand, the well-known restriction factors of head pose estimation are illumination, facial occlusion, and facial expression [33]. Once these restrictions are resolved in future research, the accuracy of the head pose estimation algorithms will improve, so will the performance of FaceLive. It is also possible to further improve FaceLive by refining the key feature parameters and the classification models.

Considering the potential of FaceLive, it might force an adversary to develop much more sophisticated attacks. For example, the adversary may obtain a suitable facial video and program a robotic arm moving a target mobile device appropriately according to the head poses in the video. Like the prior research [8, 12] relying on the same liveness indicator, FaceLive would be vulnerable. However, our technique significantly raises the bar for adversaries to perform such attacks.

7. CONCLUSION

In this paper, we propose a practical and robust liveness detection mechanism, named FaceLive, to protect face authentication from the MFF-based attacks. FaceLive can effectively detect the MFF-based attacks without requiring any additional hardware on mobile devices. FaceLive is robust to unsuccessful detection of partial facial landmarks as well as inaccurate measurement of head pose changes, which may happen during face authentication in varied environments.

8. ACKNOWLEDGMENTS

This material is based on research work supported by the Singapore National Research Foundation under NCR Award Number NRF2014NCR-NCR001-012.

9. REFERENCES

[1] A. F. Abate, M. Nappi, D. Riccio, and G. Sabatino. 2d and 3d face recognition: A survey. *Pattern Recognition Letters*, 28(14):1885–1906, 2007.

[2] A. Acquisti, R. Gross, and F. Stutzman. Faces of facebook: Privacy in the age of augmented reality. *BlackHat USA*, 2011.

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

[4] R. V. Benson. *Euclidean geometry and convexity*. McGraw-Hill New York, 1966.

[5] B. Biggio, Z. Akhtar, G. Fumera, G. Marcialis, and F. Roli. Security evaluation of biometric authentication systems under real spoofing attacks. *Biometrics, IET*, 1:11–24, 2012.

[6] B. Carter. Filter design in thirty seconds. *Application Report from Texas Instruments*, 2001.

[7] M. M. Chakka, A. Anjos, S. Marcel, R. Tronci, D. Muntoni, G. Fadda, M. Pili, N. Sirena, G. Murgia, M. Ristori, et al. Competition on counter measures to 2-d facial spoofing attacks. In *IJCB 2011*, pages 1–6. IEEE, 2011.

[8] S. Chen, A. Pande, and P. Mohapatra. Sensor-assisted facial recognition: An enhanced biometric authentication system for smartphones. In *MobiSys 2014*, pages 109–122, 2014.

[9] H. Cho, R. Roberts, B. Jung, O. Choi, and S. Moon. An efficient hybrid face recognition algorithm using pca and gabor wavelets. *IJARS 2014*, 11(59):1–8, 2014.

[10] A. De Luca, A. Hang, E. von Zezschwitz, and H. Hussmann. I feel like i'm taking selfies all day!: Towards understanding biometric authentication on smartphones. In *CHI 2015*, pages 1411–1414. ACM, 2015.

[11] Facelock.mobi. http://www.facelock.mobi/facelock-for-apps.

[12] R. D. Findling and R. Mayrhofer. Towards face unlock: on the difficulty of reliably detecting faces on mobile phones. In *MoMM 2012*, pages 275–280. ACM, 2012.

[13] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine learning*, 29(2-3):131–163, 1997.

[14] M. Gardner and S. Dorling. Artificial neural networks (the multilayer perceptron)ąła review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14):2627–2636, 1998.

[15] R. Ghiass, O. Arandjelovic, H. Bendada, and X. Maldague. Infrared face recognition: A literature review. In *IJCNN 2013*, pages 1–10, 2013.

[16] Google. http://www.android.com/about/ice-cream-sandwich/.

[17] R. Gross and A. Acquisti. Information revelation and privacy in online social networks. In *WPES 2005*, pages 71–80, 2005.

[18] R. Gross, I. Matthews, J. Cohn, T. Kanade, and S. Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.

[19] J. A. Hanley and B. J. McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[20] B. K. Horn and B. G. Schunck. Determining optical flow. In *1981 Technical Symposium East*, pages 319–331. International Society for Optics and Photonics, 1981.

[21] D. W. Hosmer Jr and S. Lemeshow. *Applied logistic regression*. John Wiley & Sons, 2004.

[22] A. K. Jain, A. A. Ross, and K. Nandakumar. *Introduction to Biometrics*. Springer, 2014.

[23] 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.

[24] O. Kahm and N. Damer. 2d face liveness detection: An overview. In *BIOSIG 2012*, pages 1–12, 2012.

[25] I. Kamal. *WFR, a Dead Reckoning Robot–A Practical Application to Understand the Thoery*. IKA Logic: Electronics Solutions Online Documents, 2008.

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

[27] Lenovo. http://veriface.software.informer.com/.

[28] Y. Li, K. Xu, Q. Yan, Y. Li, and R. H. Deng. Understanding osn-based facial disclosure against face authentication systems. In *AsiaCCS 2014*, pages 413–424, 2014.

[29] J. Maatta, A. Hadid, and M. Pietikainen. Face spoofing detection from single images using micro-texture analysis. In *IJCB 2011*, pages 1–7. IEEE, 2011.

[30] P. S. Mann. *Introductory statistics*. John Wiley & Sons, 2007.

[31] R. Mayrhofer and T. Kaiser. Towards usable authentication on mobile phones: An evaluation of speaker and face recognition on off-the-shelf handsets. In *IWSSI/SPMU 2012*. Citeseer, 2012.

[32] M. Müller. Dynamic time warping. *Information retrieval for music and motion*, pages 69–84, 2007.

[33] E. Murphy-Chutorian and M. M. Trivedi. Head pose estimation in computer vision: A survey. *TPAMI*, 31(4):607–626, 2009.

[34] L. O'Gorman. Comparing passwords, tokens, and biometrics for user authentication. *Proceedings of the IEEE*, 91(12):2021–2040, 2003.

[35] S. Ohayon and E. Rivlin. Robust 3d head tracking using camera pose estimation. In *ICPR 2006*, volume 1, pages 1063–1066. IEEE, 2006.

[36] G. Pan, L. Sun, Z. Wu, and S. Lao. Eyeblink-based anti-spoofing in face recognition from a generic webcamera. In *ICCV 2007*, pages 1–8, 2007.

[37] M. Rahman, U. Topkara, and B. Carbunar. Seeing is not believing: visual verifications through liveness analysis using mobile devices. In *ACSAC 2013*, pages 239–248. ACM, 2013.

[38] J. Rice. http://www.androidpolice.com/2012/08/03/android-jelly-beans-face-unlock-liveness-check-circumvented-with-simple-photo-editing/.

[39] F. Rocca, M. Mancas, and B. Gosselin. Head pose estimation by perspective-n-point solution based on 2d markerless face tracking. In *Intelligent Technologies for Interactive Entertainment*, pages 67–76. Springer, 2014.

[40] R. K. Rowe, U. Uludag, M. Demirkus, S. Parthasaradhi, and A. K. Jain. A multispectral whole-hand biometric authentication system. In *Biometrics Symposium, 2007*, pages 1–6. IEEE, 2007.

[41] J. M. Saragih. Deformable face alignment via local measurements and global constraints. In *Deformation Models*, pages 187–207. Springer, 2013.

[42] P. Siirtola and J. Röning. Recognizing human activities user-independently on smartphones based on accelerometer data. *IJIMAI 2012*, 1(5), 2012.

[43] T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination, and expression database. *TPAMI*, 25(12):1615–1618, 2003.

[44] F. Stenger. Integration formulae based on the trapezoidal formula. *IMA Journal of Applied Mathematics*, 12(1):103–114, 1973.

[45] F. Stutzman, R. Gross, and A. Acquisti. Silent listeners: The evolution of privacy and disclosure on facebook. *Journal of Privacy and Confidentiality*, 4(2):2, 2013.

[46] J. Timmer, C. Gantert, G. Deuschl, and J. Honerkamp. Characteristics of hand tremor time series. *Biological Cybernetics*, 70(1):75–80, 1993.

[47] S. Tupling and M. Pierrynowski. Use of cardan angles to locate rigid bodies in three-dimensional space. *Medical and Biological Engineering and Computing*, 25(5):527–532, 1987.

[48] Visidon. http://www.visidon.fi/en/Home.

[49] J. Wilder, P. J. Phillips, C. Jiang, and S. Wiener. Comparison of visible and infra-red imagery for face recognition. In *FG 1996*, pages 182–187. IEEE, 1996.

[50] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfeld. Face recognition: A literature survey. *CSUR*, 35(4):399–458, 2003.