# Protecting Real-time Video Chat against Fake Facial Videos Generated by Face Reenactment

Jiacheng Shang and Jie Wu
Center for Networked Computing, Temple University, Philadelphia, PA 19121

***Abstract*—With the rapid popularity of cameras on various devices, video chat has become one of the major ways for communication, such as online meetings. However, the recent progress of face reenactment techniques enables attackers to generate fake facial videos and use others' identities. To protect video chats against fake facial videos, we propose a new defense system to significantly raise the bar for face reenactment-assisted attacks. Compared with existing works, our system has three major strengths. First, our system does not require extra hardware or intense computational resources. Second, it follows the normal video chat process and does not significantly degrade the user experience. Third, our system does not need to collect training data from attackers and new users, which means it can be quickly launched on new devices. We developed a prototype and conducted comprehensive evaluations. Experimental results show that our system can provide an average true acceptance rate of at least $92.5\%$ for legitimate users and reject the attacker with mean accuracy of at least $94.4\%$ for a single detection.**

***Index Terms*—Face forgery, face liveness detection, real-time video chat.**

## I. Introduction

In the past a few years, thanks to fast internet speeds and the powerful processing capacity of personal electronic devices, video chat has become a major form of communication. Compared with text-based or audio-based communication, video chat enables users to observe the real emotions and activities of each other without physically being together, which makes the information delivered more accurate and the relationship establishment more efficient. Therefore, many video chat software (e.g. Skype [1] and WebEx [2]) are released for various applications, such as conference meeting, interviewing, and making friends. Based on a report from Statista, the estimated number of Skype users is expected to be 1.67 billion in 2020 [3].

There are two major channels in real-time video chat: image and audio. By default, both channels are regarded as real information since they are generated in real-time, which is why video chat is used as an alternative way to validate the identity of a user in practice. However, since a malicious user can easily get the victim's videos and voice from social networks, both channels can be well counterfeited with the development of AI-assisted techniques. For example, the recorded voice of the victim can be replayed to pass through current voice-based authentication systems. Similarly, recent research in face reenactment shows that the facial expressions on one face can be transferred to any other face in real-time. These facts enable the malicious user to easily use the victim's identity, which poses a serious threat to legitimate users. Even if the voice replay attack can be efficiently countered by using voice liveness detection techniques [4]–[7], attackers can still fool legitimate users by generating fake image channels.

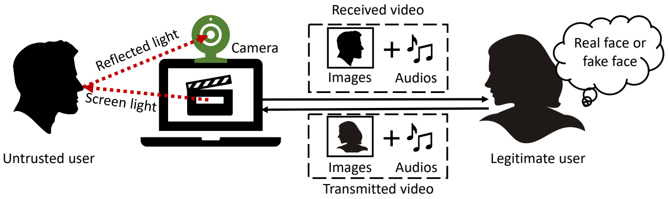


Fig. 1. Face forgery detection in real-time video chat.

To defend against fake face videos, various face liveness detection systems are designed using either artifact detection-based methods [8]–[12] or challenge-response-based methods [13], [14]. The basic assumption of the artifact detection-based method is that fake facial images must have imperfect artifact detections. By extracting proper features, the fake facial images can be detected using various classification models. However, in order to gain enough knowledge for building a robust classifier, artifact detection-based methods have to collect fake videos in advance, which usually involves significant training. Moreover, artifact detection usually requires lots of computation resources to achieve better feature extraction and classification, which is not available on resource-limited devices. Challenge-response-based methods are based on the nature of human activities. For example, FaceLive can detect the media-based facial forger by correlating the head movement measured by motion sensors and head pose change recorded in videos [13]. However, the face reenactment attacker can still easily break FaceLive by faking the sensor data since it can have enough knowledge of the target video. Moreover, since the detection is done on the attacker side, the attacker can even send the legitimate user a wrong detection result. Recently, Tang et al. [14] proposed a new liveness detection method by randomly flashing pre-designed pictures (e.g. white and black scenes) on a screen and analyzing the face-reflected light. Nevertheless, their work also relies on a neural network for accurate classification. Moreover, the flashing pictures replace the original video frames, which will degrade the user experience between two legitimate users.

Considering the limitations of existing solutions, we propose a defense system for real-time video chat against fake facial

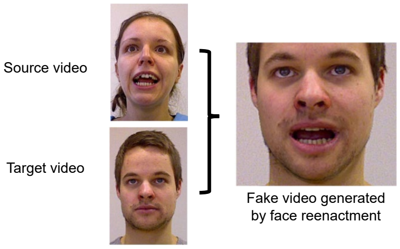


Fig. 2. An example of face reenactment techniques [16].

videos generated by face reenactment techniques. As shown in Fig. 1, our system requires no extra sensors except the screen and camera that are available on all videotelephony devices. Specifically, the screen is used to emit light signals, and the camera works as a sensor to measure the *relative luminance* (simplified as *luminance* [15] in this paper) of the lights that are reflected from the untrusted user's face. The key insight behind our system is that the luminance of the face-reflected light is proportional to that of the screen light for a legitimate user. Since the face reenactment attacker cannot generate the real-time face reflection in a photo-realistic fashion, the legitimate user can detect the face forgery by: 1) introducing luminance changes in the transmitted video by changing the area of light metering; 2) measuring the correlation between the luminance changes of the screen light and face-reflected light.

To achieve our goal, we solve three major challenges in the design of our system. The first challenge is to robustly extract the luminance information of face reflection from the videos. To address this issue, we leverage the facial landmark detection algorithm to locate the lower part of the nasal bridge as the area of interest and calculate the luminance information using only the color information within this area. Second, the luminance signals are noisy and cannot be directly used for correlation measurement. To solve this problem, we remove the noise components using signal processing techniques and extract the significant light change from filtered signals. The last challenge is to extract useful features from the filtered signal and build a classifier for robust and accurate detection. In our system, we extract four features that describe the luminance change behavior and trend from the filtered signal. A local outlier factor-based classifier is trained on selected features for the final decision.

Compared with existing works, our system has three major strengths: 1) *low-cost*: our system does not require extra hardware or intense computation resources; 2) *good user experience*: since the luminance change in the transmitted video is made by controlling the exposure level, both users can still see each other's faces with only limited loss of video information; and 3) *zero training effort*: our system does not need to collect training data from either a new user or attackers, which means our system can be quickly launched on new devices. We summarize our contributions as follows:

- This is the first work where the luminance of face-reflected light is used to defend against face reenactment attackers.
- We propose robust solutions for extracting luminance signals from videos and finding significant luminance changes from noisy luminance signals.
- We extract four strong features from the filtered signals to describe the luminance change behavior and trend. Moreover, we propose a local outlier factor-based classifier to detect fake faces in videos without collecting training data from either a new user or attackers.
- We develop a prototype and conduct comprehensive evaluations. Experimental results show that our system can provide an average true acceptance rate of at least $92.5\%$ for legitimate users and reject face reenactment attackers with mean accuracy of at least $94.4\%$ for each detection.

## II. Preliminary

### A. Face Forgery using Face Reenactment

Deepfake (a portmanteau of "deep learning" and "fake") is a type of techniques for human image synthesis based on artificial intelligence. It is used to combine and superimpose existing images and videos onto source images or videos using machine learning techniques . Because of these capabilities, Deepfake techniques have been used to create fake celebrity pornographic videos, fake news, and malicious hoaxes. Face reenactment is an example of Deepfake techniques. The goal of face reenactment techniques is to animate the facial expressions of the target video by a source actor and re-render the manipulated output video in a photo-realistic fashion. Fig. 2 shows an example of the face reenactment technique reported in [16]. We can see that the facial expression in the source video is transferred to the person in the target video with high quality. Compared with other real-time face forgery techniques (e.g. face swapping), face reenactment creates fewer artifacts while achieving high frame rates (up to 47.5 Hz in [17]). For a legitimate user in video chat scenarios, it is hard to detect face reenactment attacks with high accuracy. Although face reenactment techniques have made great success on face forgery, their nature also gives us the insight to defend against them. Since face reenactment techniques only focus on transferring the facial expression, the luminance change of the output video is the same as the target video, which means the attacker cannot have the correlated luminance change of face-reflected light. Even if the face reenactment attacker can use the source actor to observe the luminance change and generate the change in the output video, the extra computational overhead will largely reduce the frame rate make real-time attacks unfeasible.

### B. Light Metering of Digital Cameras

To achieve consistent and accurate exposures in the recorded videos, the light meter is essential for current digital cameras. In general, the camera controls the shutter speed and aperture by predicting how much light is actually hitting the subject. Current cameras provide users with various ways to meter light. Among them, spot and multi-zone metering modes are most used and widely available. In multi-zone metering, the camera measures the light intensity at multiple points in the

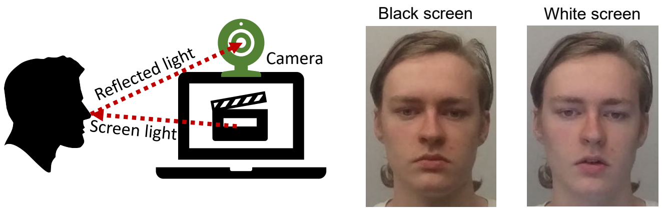


Fig. 3. Luminance of face-reflected light when screen light changes.

scene and then combines the results to find the setting for the ideal exposure. Therefore, multi-zone metering can produce balanced exposure for most scenes and is used as the default mode for most cameras. Alternatively, with spot metering, the camera will measure only a very small area of the scene. By default, this small area is at the center of the scene, but the user or application can easily select a different off-centre spot. If the spot is moving from a relatively low-luminance area to a high-luminance area, the camera will let less light in, which leads to a diminished brightness in the darker area. Similarly, if the spot is moving from a relatively high-luminance area to a low-luminance area, the luminance of the scene rises. Hence, by moving the metering spot between high-luminance and low-luminance areas, the legitimate user can easily control the overall luminance of its video. Since the exposure only changes the brightness of each pixel, this method can reserve partial information (e.g. the face of the legitimate user) in the scene, which ensures a certain level of user experience.

### C. Face Reflection of Screen Light

When the untrusted user watches the legitimate user's facial video, the camera can capture the screen light that is reflected by the face of the untrusted user. Here, we model the face reflection of screen light based on the Von Kries coefficient law [18]. For a single type camera, a diagonal model can be described as:

$$I_c(x) = E_c(x) \times R_c(x),\ c \in \{R, G, B\}, \tag{1}$$

where $x$ is a pixel on the face, $c$ is the light with different colors (red, green, and blue), $I_c$ is the luminance of corresponding color, $E_c$ is the illuminant spectral power distribution of the screen light on $x$, and $R_c$ is the reflectance of pixel $x$. Therefore, if we focus on a pixel with the same reflectance and change the light luminance, then we have:

$$\frac{I_c(x)'}{I_c(x)} = \frac{E_c(x)'}{E_c(x)},\ c \in \{R, G, B\}, \tag{2}$$

where $I_c(x)'$ and $E_c(x)'$ are the luminance and illuminant spectral power distribution after the change of screen light. From this equation, we can observe that the luminance of the face reflection is proportional to that of the screen light, which serves as the basic insight of our system.

### D. Feasibility Study

To achieve our goal, we first show that the luminance of the face-reflected light is highly correlated to that of the screen light. Specifically, we made a video that flashes between white and black with a frequency of 0.2 Hz and displayed this video on a Dell 27-inch Light-emitting diode (LED) Monitor. We asked a volunteer to sit in front of the monitor while using the front camera of an iPhone 7 to record his facial video. During the recording, the volunteer can freely move the head as long as the whole face can be captured by the camera. Fig. 3 shows the faces when the screen shows black and white colors, respectively. We can clearly observe that the luminance of the face-reflected light increases when the color changes from black to white. As a reference, the luminance value of the nasal bridge increases from around 105 to around 132. Moreover, this fact is true for all types of screens including LED, liquid crystal display (LCD), and organic LED (OLED) since they all reduce the amount of emitted light when displaying darker scenes. This simple case implies that the luminance of the face-reflected light does change proportionally to that of the screen light, which shows the possibility of detecting fake faces using the correlation between two luminance signals.

### E. Challenges

Although we can observe the corresponding luminance change of the face-reflected light while the screen's color changes between black and white, it is still challenging to apply this insight to real video chat scenarios for fake forgery detection. First, the face of the untrusted user will likely be moving in the scene and can be partially occluded by other objects (e.g. hair and sunglasses), which introduces extra noise to the luminance signals of the face-reflected light. To address this issue, our system only extracts the luminance information from the lower part of the nasal bridge since this area can be robustly located using the facial landmark detection algorithm and is the least likely part to be obfuscated.

The second challenge is to obtain the luminance change information from the noisy luminance measurements. The raw luminance signals contain various types of noise. For example, dynamic scenes in the video will introduce high-frequency noise to the raw luminance signal of the screen light. Additionally, the luminance change is weaker in practice than in the ideal case in a feasibility study. To remove the noise and robustly locate each luminance change, we designed a series of filters and apply them on the raw luminance signals in order.

The last challenge is to extract useful features from the filtered signal and build a classifier for robust and accurate face forgery detection. To solve this problem, we select four features that describe when and how the luminance signal significantly changes. To reduce the training cost while still ensuring good performance, we build a strong classifier without collecting training data from the attacker and new users using the local outlier factor model.

## III. System Design

### A. Adversary Model

In our adversary model, the attacker aims to impersonate others using face reenactment while video chatting with vic-

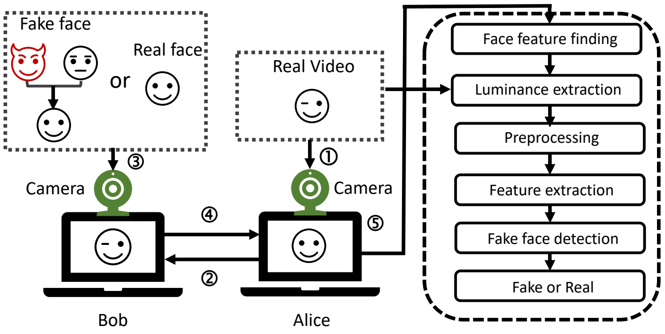


Fig. 4. System architecture.

tims. The capability of the face reenactment attacker is limited in the sense that: 1) the attacker has already or is able to set up a video chat connection with the victim; 2) the attacker can generate fake facial videos with high quality in real-time using any face reenactment technique; and 3) the attacker can redirect the input stream of the current video chat software (e.g. Skype) from the camera to the fake facial videos using a virtual web camera. When these tasks are performed, we suppose the victim cannot visibly identify the fake facial video as a forgery. Note that the adversary model we considered is much stronger than traditional models where attackers replay the fake facial videos using another screen because the facial videos are directly fed to video software without any loss and interference.

The objective of our system is to significantly raise the bar for such face reenactment attacks. To break our defense system, the attacker needs to reconstruct the face-reflected light on the fake face with high quality based on the relative locations of the head, camera, and the screen in real-time. For this, the attacker has to: 1) introduce an extra image processing layer for each frame to reconstruct the face-reflected light; and 2) have enough computation resources to ensure the real-time attack. Therefore, our system is difficult to attack.

### B. System Architecture

The key idea underlying our system is to measure the luminance correlation between the screen light and the face-reflected light. When a legitimate user is using videotelephony with an untrusted user, the camera can capture the screen light that is reflected by the untrusted user's face. By comparing the luminance of the screen light and the face-reflected light, we can determine if the face is from a real person or generated by face reenactment techniques. There are two major phases for using our system: a training phase and a detection phase. In the training phase, our system will learn the decision strategy based on the knowledge in the legitimate users' data. After that, our system is ready to be used for detection. Our detection methods can be triggered multiple times during the real-time video chat. If the untrusted user is detected as an attacker, an alert will be sent to the legitimate user to avoid further loss.

Fig. 4 shows the detailed process of our system in five steps. A legitimate user Alice wants to validate whether the facial video sent from the untrusted user Bob is real or fake. To do this, Alice records her own facial videos using a camera in step 1 and sends the real-time facial video via the internet to Bob in step 2. On Bob's side, his device receives Alice's video and displays it on his screen, which means that the luminance of the screen light largely depends on the content in Alice's video in real-time. At the same time, as illustrated in step 3, Bob is recording his facial video whose luminance change should be influenced by not only the ambient light in Bob's environment but also his screen light. By receiving Bob's video in step 4, Alice can get the luminance information of both Bob's screen light and Bob's face-reflected light. In our system running on Alice's device, we first extract the luminance information in both videos and apply filters to the raw signals to extract only significant light changes.

## IV. Luminance Extraction

The goal of our system is to detect the liveness of the face in the video by measuring the correlation between luminance signals of the screen light and face-reflected light. Therefore, we first need to robustly extract these two types of luminance information from the two videos. Since we are only interested in the overall luminance of the screen light, we first compress each frame of the transmitted video into a single pixel, and use the luminance value of the compressed pixel to represent the overall luminance of the transmitted video. The luminance of a pixel is defined as:

$$C = 0.2126R + 0.7152G + 0.722B, \tag{3}$$

where $C$ is the luminance value calculated using linear Red Green Blue (RGB) values. The coefficient of each color is assigned based on the human visual perception of brightness.

However, not all facial parts can be used to measure luminance changes. For example, the user may blink the eyes or talk during the recording. Such activities will introduce a lot of variances between neighboring frames. Also, users may wear glasses that reflect lights from other sources, which will introduce much noise to the luminance measurements. Based on our preliminary study, we find that the lower part of the nasal bridge has the most stable images and is hard to be occluded in most cases. Moreover, the luminance changes caused by different screen lights at this area are easy to detect. Therefore, we extract only the lower part of the nasal bridge from each frame of the video for luminance measurement.

When a legitimate user receives the video from the untrusted user, our system extracts frames with a sample rate of 10 Hz. For each frame, we detect the location of the lower part of the bridge by using a facial recognition API for Python [19]. As shown in Fig. 5, the facial recognition API can report four locations on the nasal bridge and five locations on the nasal tip. Since the sampled frames can vary in size depending on camera hardware, we use the locations of the nasal bridge and nasal tip to extract the interested area. As shown in Fig. 5, given the coordinates of the nasal bridge $(a_1, b_1)$ and nasal tip $(a_2, b_2)$, the side length of the interested area is $l = |b_1 - b_2|$.

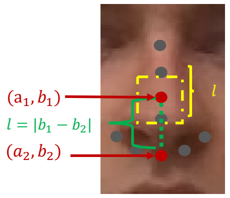


Fig. 5. Facial feature localization and interested area extraction.

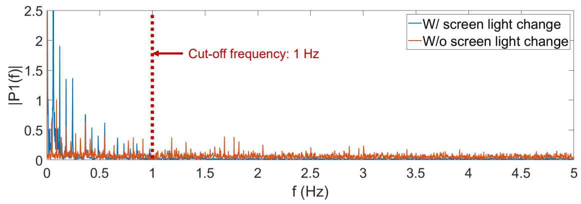


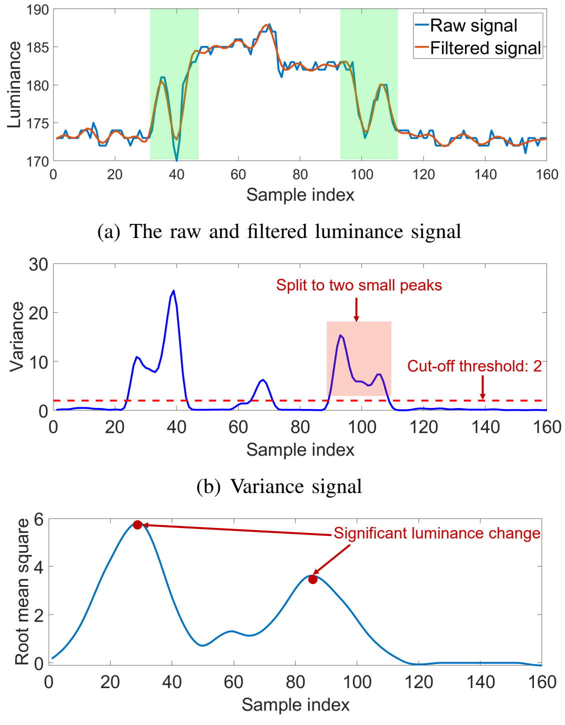
Fig. 6. The spectrum of luminance signals w/ and w/o screen light change.

A square whose center is $(a_1, b_1)$ is extracted from the frame to calculate the luminance. We use the same methods to get the luminance information from the area of interest. Fig. 7(a) shows the luminance signal generated from the lower part of the nasal bridge, and we can see significant rising edge and falling edge appear when the luminance of the screen light significantly changes (green areas).

## V. Preprocessing

As shown in Fig. 7(a), the raw luminance signals contain various noise. For the transmitted video, the noise is mainly from the object movement in the scene. For the face reflection in the received video, the noise can be introduced by external light sources. Moreover, the inaccurate face localization can lead to jittering in the interested area, which further influences the luminance extraction of the face reflection. Hence, the raw luminance signals need to be filtered before being used for feature extraction.

Fig. 6 illustrates the spectrum of the luminance signals of the face-reflected light. It is clear that high-frequency noise exists across whole frequency bands, while screen light changes influence the luminance of the face-reflected light with low frequency under 1 Hz. Based on this observation, we first use a low-pass filter with a cut-off frequency of 1 Hz. As shown in Fig. 7(a), most high-frequency components are removed while the overall trend is reserved. In our system, we only consider significant luminance changes in both luminance signals for two reasons. First, only significant luminance changes in the transmitted video can generate luminance changes in the interested area of the received video. Second, the significant luminance changes in the received video are robust to noise and easier to detect. However, it is hard to locate each significant luminance change in the filtered signal since low-frequency noise still exists. To locate all significant light change in the filtered signal, we leverage a moving window with length of 10 samples and calculate the short-time variance within each window. The basic insight is that the low-frequency noise within a window only generates a low variance. Moreover, the variance value in the moving window can reach a local maxima in two cases: 1) the luminance rapidly increases to a high value; and 2) the luminance drops to a much lower value. Therefore, each significant luminance change can be located by finding the local maxima in the variance signal.



(a) The raw and filtered luminance signal

(b) Variance signal

(c) Smoothed variance signal

Fig. 7. Preprocessing of luminance signals.

Nevertheless, the variance signal cannot be directly used for locating significant light change. As shown in Fig. 7(b), low-frequency noise can either generate small spikes in the variance signal or split a significant luminance change into multiple lower, neighboring peaks. To remove small spikes, we apply a threshold filter on the variance signal with a cut-off threshold of 2. To group neighboring lower peaks into one significant luminance change, we further smooth the variance signal by applying a moving window with a length of 30 samples and calculating the root-mean-square value in each window. Then, we leverage a Savitzky-Golay filter [20] with a window length of 31 samples using polynomial fitting and a moving average filter with a window length of 10 samples to further smooth the signal, and the result is shown in Fig. 7(c). Finally, the traditional peak finding algorithm is applied on each smoothed variance signal respectively. Since the luminance variation range of the screen light is much larger than that of the face-reflected light, the minimal prominence of the peaks is set to 10 and 0.5 for the screen light and face-reflected light, respectively.

## VI. Feature Extraction

In order to detect a fake face in the video, we need to extract proper features that can describe the correlation between two relative luminance signals. In our system, we consider both the similarities of luminance change behaviors and the correlation of luminance change trends. The luminance change behavior is a vector where the value of each element is the time

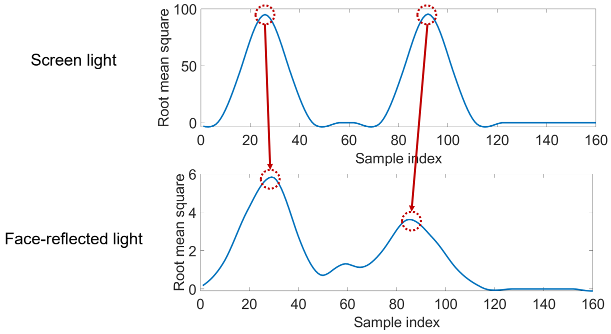


Fig. 8. The luminance change in transmitted and received video respectively.

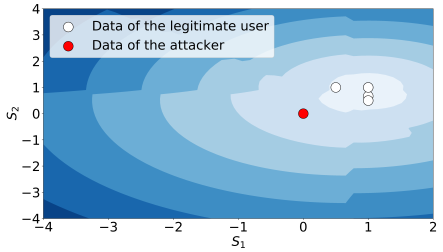


Fig. 9. An example of LOF-based classification.

when a significant luminance change happens. Therefore, the luminance change behavior focuses on the timestamps when significant luminance changes happen while ignoring the trend of the signal. The luminance change trend is the smoothed variance signal after preprocessing and is used to describe how the luminance changes over time.

*1) Luminance change behavior:* If both luminance signals are legitimate, there is a strong correlation between them. In other words, for any significant luminance change in one signal, we can always find a matched luminance change in another one. To quantitatively describe how similar two luminance change behaviors are, we define two behavior similarity metrics $z_1$ and $z_2$. The proportion of matched luminance changes in the transmitted video $z_1$ is defined as:

$$z_1 = \frac{1}{N} \times F(T, R), \tag{4}$$

where $N$ is the number of significant luminance changes in the transmitted video, $T$ is the preprocessed luminance signal of the transmitted video, $R$ is the preprocessed luminance signal of the received video, and $F(T, R)$ is a function whose output value is the number of matched luminance changes in the transmitted video. Similarly, the proportion of the matched luminance change in the received video $z_2$ is defined as:

$$z_2 = \frac{1}{M} \times G(T, R), \tag{5}$$

where $M$ is the number of significant luminance changes in the received video and $G(T, R)$ is a function whose output value is the number of matched luminance changes in the received video. For a legitimate user, both values are expected to be 1 or very close to 1, while the values of an attacker should be close to 0. Fig. 8 shows two luminance signals collected from a legitimate user. It is clear that, for each luminance change in one signal, we can always find a matched luminance change in another one, which means both $z_1$ and $z_2$ are equal to 1.

*2) Luminance change trend:* In the luminance change behavior, we only consider when significant luminance changes happen while ignoring the trend of the signal. In the worst case, the attacker's signal can have the same luminance change behavior but with considerably different shapes of luminance signals. Therefore, besides considering the similarity between two luminance change behaviors, we also evaluate the correlation of their trends. To remove the mismatching introduced by network delay, we first estimate and remove the delay based on the average time difference between matched luminance changes. Since we only consider the trend of the luminance signal instead of absolute values, we further normalize each smoothed variance signal to $[0, 1]$. Then, each signal is cut into two segments with equal length. For each pair of segments of two signals, we leverage Pearson correlation coefficient [21] to measure the correlation of their trends. Specifically, the correlation coefficient $corr(x, y)$ between a pair of signal segments is defined as:

$$corr(X, Y) = \frac{1}{L} \sum_{i=1}^{L} (\frac{x_i - \overline{X}}{\sigma_X})(\frac{y_i - \overline{Y}}{\sigma_Y}), \tag{6}$$

where $L$ is the number of segments, $X = (x_1, x_2, \ldots, x_L)$ and $Y = (y_1, y_2, \ldots, y_L)$ are a pair of normalized signal segments with length of $L$, $\overline{x}$ is the mean value of $x$, $\overline{y}$ is the mean value of $y$, $\sigma$ represents the standard deviation. The value range of $corr(x, y)$ is $[-1, 1]$. Ideally, $corr(x, y)$ should be 1 if two smoothed variance signals are positively correlated. In other words, the larger the $corr(x, y)$, the more positive correlation exists between two smoothed variance signals. Since we have two correlation coefficients calculated from two pairs of segments, we only use the smaller one of them as the third feature $z_3$. Besides, we also use the maximum dynamic time warping (DTW) distance (expressed with $z_4$) between each pair of segments as the fourth feature to describe the correlation of luminance change trends. Since the range of $z_4$ is much larger than the other three features, we divide it by 30 to reduce its influence in the classification.

## VII. Fake Video Detection

### A. Fake video detection for a single video clip

Although we can simply check whether a luminance change happens at the same time in both videos, it will make a weak luminance change in one video be identical to a strong luminance change in another one, which increases the chance of attackers to pass the check. Therefore, we also need to measure how well two luminance signals match each other via classification techniques. To do the classification, a naive idea is to collect training data from both legitimate users and face reenactment attackers. However, it will involve much training cost to collect data from every new legitimate user. Moreover, it is even harder to get data from all possible face reenactment attackers. Therefore, we need to build a classifier with good classification performance using only the data of a limited number of legitimate users. In our system, we build a strong classifier using the local outlier factor (LOF) model [22] since it has good performance and fewer parameter adjustment requirements. Specifically, the dataset sent to the LOF model consists of two parts: the dataset collected from legitimate users and one new data from the untrusted users. Since the attacker's features are distinct from those of legitimate users on at least one dimension, the attacker's data appears as an outlier in the whole dataset.

Given a feature vector $z = [z_1, z_2, \ldots, z_K]$ of the untrusted user's data, the local reachability density (LRD) of a feature vector $z$ is defined as:

$$LRD(z) = 1/(\frac{\sum_{r\in N_k(z)} max\{k\text{-}dis(r), d(z,r)\}}{|N_k(z)|}), \quad (7)$$

where $N_k(z)$ are the $k$ nearest neighbors (legitimate users' data), $r$ is a legitimate user's data that is also the $k$ nearest neighbors of $z$, $k\text{-}dis(r)$ is the distance from the object $r$ to the $k^{th}$ nearest neighbor, and $d(z,r)$ is the euclidean distance between feature vectors $z$ and $r$ on the feature hyperplane. LOF model determines whether the signal is from an attacker based on comparing the local densities of $z$ and its $k$-nearest neighbors using

$$LOF_k(z) = \frac{\sum_{r\in N_k(z)} \frac{LRD(r)}{LRD(z)}}{|N_k(z)|}. \quad (8)$$

Since the attacker's features are distinct from those of legitimate users on at least one feature dimension, so the attacker's data point should be away from the cluster for legitimate users, which means its values of $LOF_k(z)$ are larger than 1 on the feature hyperplane. Based on this observation, our system determines whether the signal is generated by the attacker by setting a threshold $\tau$. If the value of $LOF_k(z)$ is larger than $\tau$, an attacker is claimed to be detected. Fig. 9 illustrates an example of LOF-based classification using two features $z_1$ and $z_2$. The darkness of the background represents the value of $LOF_k(z)$. The darker the background is, the larger the $LOF_k(z)$ is. We can observe that the $LOF_k(z)$ values of legitimate users are all less than 1.5, while that of the attacker is 2. By setting a threshold $\tau = 1.8$, the attacker can be accurately detected. In our system, the decision threshold $\tau$ is set to 3, and the number of neighbors is set to 5.

### B. Decision combination for multiple rounds of detection

Since our solution does not require intense computational resources, it is possible to trigger our system multiple times during the video chat to tolerate single wrong classification. To combine the detection results of multiple attempts, we involve them in a majority voting game where each player has equal weight. Considering the final result is produced based on $D$ detection attempts, an untrusted user is regarded as a face reenactment attacker if its votes exceed $0.7 \times D$. The coefficient 0.7 is determined based on the detection accuracy of each single detection, which is reported in Section VIII-C.

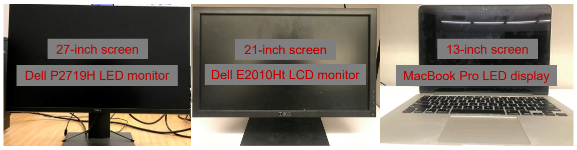


Fig. 10. Monitors used in our experiments.

## VIII. Evaluation

### A. Implementation and Dataset

Like the video chat scenario, our system consists of two components: a legitimate user (Alice in Fig. 4) who triggers the detection and an untrusted user (Bob in Fig. 4) with unknown legality. We implemented our testbed using a Dell 27-inch LED monitor with 85% brightness to display the video from the legitimate user. For the untrusted user who is also legitimate, we used a Google Nexus 6 smartphone to act as the camera for recording facial videos. For the untrusted user who is a face reenactment attacker, we first collect its facial videos using a Google Nexus 6 smartphone. The recorded facial videos are then fed to the driving model of ICface [23] for generating fake facial videos. The reason we use ICface is that it generated the most visually convincing results of any open-source facial reenactment method. In total, ten volunteers (four females and six males) with diverse skin colors (both dark skin and light skin) are involved in our experiments. To simulate the behavior of the legitimate user, we asked volunteers to record their daily video chat while changing the metering area by touching the smartphone screen. The collected facial videos were segmented into clips with equal length of 15 seconds. For the behavior of the untrusted user, we asked ten volunteers to act as both a legitimate user and a face reenactment attacker, respectively. For each role of each user, we replayed 40 video clips to them. For data analysis and processing, the data was then transmitted to a desktop computer with Intel(R) i7-8700 @ 3.2 GHz CPU and 32 GB of RAM.

### B. Evaluation Metrics

To evaluate the performance of our system, we use four metrics as follow: 1) *true acceptance rate* is used to describe how accurately our system can accept a legitimate user, and it is defined as the number of accepted attempts by the number of total attempts of a legitimate user; 2) *true rejection rate* represents how accurately our system can reject an attacker. It is calculated by dividing the number of rejected attempts by the number of total attempts of an attacker; 3) *false acceptance rate* is the measure of the likelihood that the system will incorrectly accept an attacker; and 4) *false rejection rate* is the measure of the likelihood that the system will incorrectly reject a legitimate user. By combining the false acceptance rate and false rejection rate, we can also get the equal error rate of the system.

### C. Overall Performance

*1) System performance for legitimate user:* A good face forgery system should provide high usability for legitimate users, which means the true acceptance rate should be as

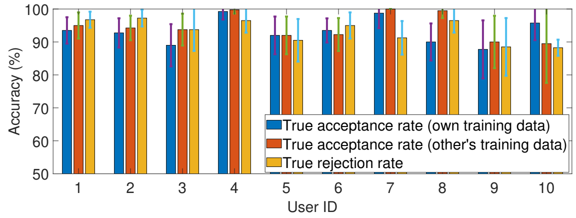


Fig. 11. Overall performance for a single detection.

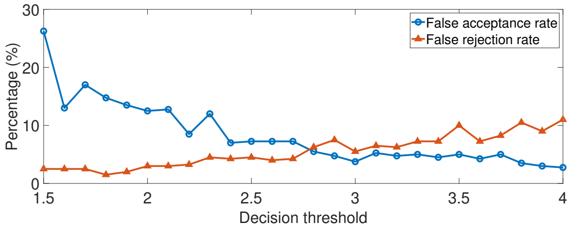


Fig. 12. Impact of decision threshold.

high as possible. To examine whether our system can be trained without collecting data from new users, we trained two classifiers using each volunteer's own data and another volunteer's data, respectively. For each volunteer, we repeated this experiment for 20 rounds to obtain the average true acceptance rate. Within each round, we randomly picked 20 instances for training and tested the system using the other 20 instances. Fig. 11 shows the true acceptance rates for a single detection attempt. We can observe that our system can provide an average true acceptance rate of 92.5% when the classifier is trained using each volunteer's own data. Even if the classifier is trained using others' data, our system can still achieve an average true acceptance rate of 92.8%, which implies that our system can be quickly launched for new users without training in practice. Also, we found that the system performance of using others' training data is better than that of using own data for some users. The main reason is that the training data of these users on the feature hyperplane distribute across a larger area compared with those of other users. Since our system determines the liveness by checking the local reachability density based on a fixed threshold, gathering more training data will provide us with better classification performance.

*2) System performance against attacker:* In this experiment, we randomly picked 20 instances from each volunteer as training data, and evaluated the true rejection rate of a single detection attempt against fake facial videos generated by ICFace. As illustrated in Fig. 11, our system can successfully reject the face reenactment attacker with average accuracy of 94.4%. For some volunteers (e.g. user 2), the mean true rejection rate can reach 97.25%, which means that our system can already provide high-security protection with only one detection attempt. Although our system can still make the wrong classification with a low possibility, our decision combination strategy can tolerate a single mistake and further improve system accuracy and robustness, which will be discussed in Section VIII-F.

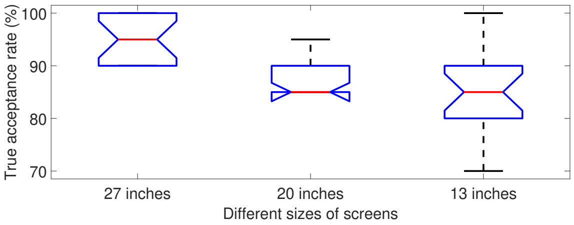


Fig. 13. True acceptance rates under different screen sizes.

## *D. Influence of Decision Threshold*

The value of the decision threshold greatly influences the system performance. If the decision threshold is too high, the legitimate users can all be passed by our system, but many attackers can also be missed. If we improve the security level by setting the decision threshold to a small value, many legitimate users will be rejected, which largely reduces usability. In this experiment, we evaluate the proper value of the decision threshold. We adjusted the decision threshold from 1.5 to 4 while using 20 randomly-picked instances for training. Fig. 12 illustrates the mean false acceptance rate and false rejection rate for different values of the decision threshold. When the decision threshold is between 2.8 and 3, our system achieved a balanced false acceptance rate and false rejection rate, which means the equal error rate of our system is about 5.5%. Therefore, we set the default value of the threshold to 3 in our system.

## *E. Influence of Screen Size*

The performance of our system largely relies on the amount of light emitted from the screen. If the luminance of the screen light is low, the luminance change of the face-reflected light would be lower due to the light scattering. In this experiment, we evaluated the system performance by using screens with different sizes shown in Fig. 10, and the results are shown in Fig. 13. Better system performance is achieved by using a larger screen, which is in line with our expectations. Even with the smallest screen in our testbed, our system can still achieve an average true acceptance rate of about 85% for a single detection. We also evaluate the system performance on a 6-inch smartphone screen. Experimental results show that our system can achieve similar performance only when the user's face is very close (about 10 cm) to the screen. When the screen is moved too far away, the luminance of the screen light is not strong enough to generate significant luminance change on the user's face.

## *F. Influence of Number of Detection Attempts*

Due to the influence of noise, our system may wrongly accept an attacker or reject a legitimate user with a low possibility. To tolerate the wrong detection for a single video clip, our system combines the results from multiple detection attempts through a majority voting procedure. Fig. 14 shows the system performance under a different number of detection attempts when 20 instances are used for training. We can observe that both true acceptance rate and true rejection rate

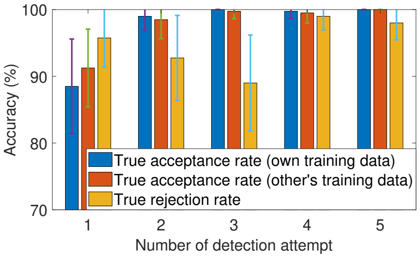


Fig. 14. Impact of the number of detection attempts.

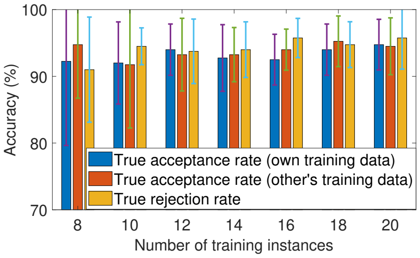


Fig. 15. Impact of the number of training instances.

are significantly improved by combining multiple detection results whether the classifier is trained using the user's own data or others' data. Moreover, the variances of accuracy (both true acceptance and true rejection rate) are largely reduced, which means the system robustness is improved by considering multiple detection attempts. Although the true rejection rates drop by at most $5\%$ for two and three attempts, it is mainly because we randomly selected training data for each classification.

### G. Influence of Number of Training Data

To quickly launch our system in practice, we want to reduce the training cost by as much as possible even if our system is trained using others' data. Therefore, we performed an experiment using the data collected from one volunteer to evaluate how many training instances are needed for good system performance, and the results are shown in Fig. 15. When the classifier is trained with eight instances, our system can already provide an average true acceptance rate of $92.25\%$ for normal users and an average true rejection rate of $91\%$ for attackers. By involving up to 20 training instances, the average true acceptance rate and true rejection rate are slightly raised to $94.75\%$ and $95.75\%$, respectively. Additionally, the standard deviations of both the true acceptance rate and true rejection rate are largely reduced by up to $8.8\%$, which shows the system robustness is largely raised by having more knowledge about the legitimate users' data.

### H. Influence of Sampling Rate

The computation overhead of our system largely depends on the frequency that we sample each video at. High sampling rates can provide us with more information about the luminance change, but the image and signal processing overhead are also multiplied. To find the lowest viable sampling rate for our system, we collected data from one volunteer and varied the sampling rate among 5 Hz, 8 Hz, and 10 Hz. As illustrated in Fig. 16, our system can still provide a mean detection accuracy of at least $95.25\%$ for both legitimate users and attackers when the sampling rate is only 8 Hz. When the sampling rate drops to 5 Hz, the mean true acceptance rate slightly decreases to about $86\%$, while the mean true rejection rate rapidly drops to only $48\%$. Therefore, our system requires a sampling rate of at least 8Hz to ensure security. We will show that most video chat devices can handle the computation overhead under a sampling rate of 8 Hz in Section IX.

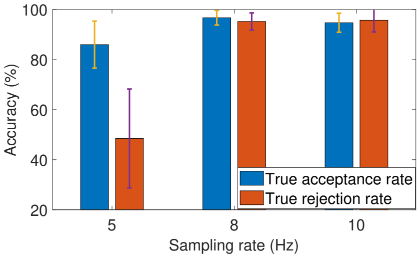


Fig. 16. Impact of the sampling rate.

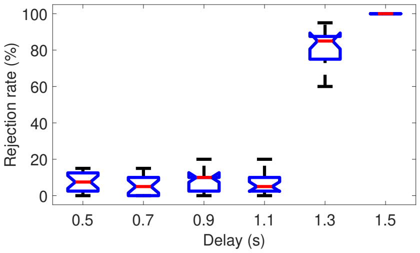


Fig. 17. The rejection rates when relative luminance signal is delayed.

### I. Influence of Ambient Light

In practice, the relative luminance of the reflected light is always under the impact of the ambient light. If the ambient light is weak, the relative luminance of the reflected light large depends on the luminance of the screen light. If the ambient light is strong, the relative luminance change of the reflected light is dominated by the ambient light instead of the screen light. In this subsection, we further evaluate the system performance under different light conditions. Experimental results show that our system can achieve similar performance that is reported in Section VIII-C. Also, the true acceptance rate of a single detection attempt drops to about $80\%$ when the illuminance on the face is increased to 240 lux (the corresponding illuminance under the light source is about 350 lux). Considering 350 lux is bright enough in an indoor environment in practice, our system can still achieve high performance in most scenarios by combining the detection results of multiple attempts.

### J. Effectiveness of The Defense System

To break our system, the attacker needs to ensure at least two things. First, the attacker has to forge the luminance change on the generate fake facial videos. Second, the synthetic luminance change should be real enough to fool the legitimate user who acts as an observer. In real attacker scenarios, it is reasonable to assume that the computation resources of the attacker are equal to the face reenactment attacker and cannot be further upgraded. Therefore, the attacker needs to more time to generate the new facial video with synthetic luminance change. Even if this generation process is feasible, it introduces a delay to the relative luminance signal. In this subsection, we evaluated the impact of this delay on the rejection rate even if the attacker can generate exactly the same relative luminance change in the fake facial video. Specifically, we shifted the relative luminance signals of a legitimate user by different delays and checked how the system performance degrades with the increases of signal delay, and the results are shown in Fig. 17. We can see that the rejection rate quickly rises to about 80% when the delay is 1.3 seconds, which means that the attacker still cannot fool our system if its forgery processing time is longer than 1.3 seconds. Considering most face reenactment techniques themselves cannot achieve this low processing time, we can argue that a strong attacker who can even forge the relative luminance signal still cannot pass our system.

## IX. Discussion

Since we target the fake face detection for real-time video chat, the computation overhead should be minimized for timely detection results. In our system, the computation overhead mainly comes from three parts: facial landmark detection for obtaining the location of the nasal bridge, feature extraction, and classification. For the facial landmark detection algorithm, recent research shows that it can be run at 300 fps on a mobile phone [24]. Besides, even with simple implementation using Matlab and Python, the feature extraction and classification can be quickly processed together within 0.2 seconds for a luminance signal extracted from a 15-second facial video. These facts imply the feasibility that our system can be implemented on more resource-limited devices (e.g. smartphones) with a video sampling rate of 10 Hz.

As a starting point, our system is evaluated in a relatively stable indoor environment. However, in practical deployment, more environmental factors need to be taken into consideration and may affect system performance. Additionally, current evaluations are performed based on a limited number of volunteers and a limited period. In the future, we need to continue the evaluation with more/diverse population samples, longer periods, and more influential factors to improve the robustness of the system.

## X. Related Work

### A. Face Reenactment Techniques

Face reenactment is a group of techniques that can transfer facial expressions from a source face to a target face. Traditionally, the work of face reenactment is done offline due to high computation resource required [17], [25]–[33]. For example, Garrido et al. [25] proposed a system that can transfer facial expressions when both the source and target faces are from the same person. They further improved their work to transfer facial expressions among different people in [26]. Recently, there has been research supporting online face reenactment, opening this technology up to a wider range of applications. A famous work called Face2Face [16] is proposed to achieve online face reenactment with about 27.6 frames per second. This fact implies that face reenactment techniques can be executed during real-time video recording, which largely improves attackers' capability for launching face forgery attack in real-time video chat.

### B. Face Liveness Detection in Videos

Considering the serious threats introduced by fake facial videos, various systems are proposed to detect fake faces in videos, which are also referred to as face liveness detection. Overall, current fake face detection methods can be grouped into two categories: artifact detection-based and challenge-response-based. Artifact detection-based methods aim to exploit artifacts that are introduced during the synthesis process using both low-level and high-level features [8]–[12], [34], such as illuminance distribution, texture, shape cues, and eye blinking. Carvalho et al. [35] proposed a forgery detection model by exploiting subtle inconsistencies in the color of the illumination of images. To further improve the performance during feature extraction, recent works propose to use deep network to capture more inconsistencies in fake images using both low-level and high-level features [8]–[12], [34]. For example, Raghavendra et al. [10] proposed a novel approach leveraging the transferable features from a pre-trained Deep Convolutional Neural Networks (D-CNN) to detect both digital and print-scanned morphed face image. Also, researchers focus on having a generalized model that can detect multiple types of face forgery. For example, a compact model is proposed in [8] to detect fake faces generated by either Face2Face [16] or Deepfakes. However, all existing artifact detection-based methods have two major limitations. First, they need to collect enough training from target face forgery techniques, which is usually expensive and hard to satisfy in practice, particularly for unknown techniques. Second, the detection procedure requires intensive computational resources (e.g. a graphics processing unit), which is not suitable for battery-limited devices.

Different from artifact detection-based methods, challenge-response-based methods leverage the nature of human activities (e.g. head movement [13]). However, the face reenactment attacker can still easily break FaceLive by faking the data of motion sensors in advance since it can have enough knowledge of the target video. Moreover, since the detection is done on the attacker's end, the attacker can even send the legitimate user a wrong detection result. A recent work detects a fake face during face authentication by randomly flashing well-designed pictures on a screen [14]. Various physical characteristics including unique textual features and uneven 3D shapes are extracted from the reflected light for robust face forgery defense. However, their challenge-response-based method has to alter the displayed content on the screen, which largely influences the user experience during video chat.

## XI. Conclusion

In this paper, we propose a defense system for real-time video chat against fake facial videos generated by face reenactment techniques. The key insight behind our system is that the luminance of the face-reflected light is proportional to that of the screen light. Therefore, we can detect face forgery by measuring the correlation between the luminance signals of the screen light and the face-reflected light. Compared with existing works, our system has three major strengths. First, it does not require extra hardware or intense computational resources. Second, our system does not replace original video frames and can ensure a certain level of user experience. Moreover, our system does not require the training data of attackers and new users, which means it can be quickly launched on any videotelephony device. Experimental results show that our system can provide an average true acceptance rate of at least 92.5% for legitimate users and reject face reenactment attacker with mean accuracy of at least 94.4% for a single detection.

## Acknowledgement

This research was supported in part by NSF grants CNS 1824440, CNS 1828363, CNS 1757533, CNS 1629746, CNS 1651947, and CNS 1564128.

## References

[1] "Skype." [Online]. Available: https://www.skype.com/en/

[2] "Webex." [Online]. Available: https://www.webex.com/

[3] E. Koptyug, "Number of estimated skype users registered worldwide from 2009 to 2024 (in billions)," 2019. [Online]. Available: https://www.statista.com/statistics/820384/estimated-number-skype-users-worldwide/

[4] J. Shang and J. Wu, "Enabling secure voice input on augmented reality headsets using internal body voice," in *2019 IEEE International Conference on Sensing, Communication and Networking*. IEEE, 2019.

[5] L. Zhang, S. Tan, and J. Yang, "Hearing your voice is not enough: An articulatory gesture based liveness detection for voice authentication," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017, pp. 57–71.

[6] S. Chen, K. Ren, S. Piao, C. Wang, Q. Wang, J. Weng, L. Su, and A. Mohaisen, "You can hear but you cannot steal: Defending against voice impersonation attacks on smartphones," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 183–195.

[7] J. Shang, S. Chen, and J. Wu, "Srvoice: A robust sparse representation-based liveness detection system," in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2018, pp. 291–298.

[8] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: a compact facial video forgery detection network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2018, pp. 1–7.

[9] A. Khodabakhsh, R. Ramachandra, K. Raja, P. Wasnik, and C. Busch, "Fake face detection methods: Can they be generalized?" in *2018 International Conference of the Biometrics Special Interest Group (BIOSIG)*. IEEE, 2018, pp. 1–6.

[10] R. Raghavendra, K. B. Raja, S. Venkatesh, and C. Busch, "Transferable deep-cnn features for detecting digital and print-scanned morphed face images," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2017, pp. 1822–1830.

[11] P. Zhou, X. Han, V. I. Morariu, and L. S. Davis, "Two-stream neural networks for tampered face detection," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2017, pp. 1831–1839.

[12] A. Rössler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, "Faceforensics++: Learning to detect manipulated facial images," *arXiv preprint arXiv:1901.08971*, 2019.

[13] Y. Li, Y. Li, Q. Yan, H. Kong, and R. H. Deng, "Seeing your face is not enough: An inertial sensor-based liveness detection for face authentication," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 1558–1569.

[14] D. Tang, Z. Zhou, Y. Zhang, and K. Zhang, "Face flashing: a secure liveness detection protocol based on light reflections," *arXiv preprint arXiv:1801.01949*, 2018.

[15] Wikipedia, "Relative luminance." [Online]. Available: https://en.wikipedia.org/wiki/Relative_luminance

[16] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2387–2395.

[17] J. Thies, M. Zollhöfer, C. Theobalt, M. Stamminger, and M. Nießner, "Headon: Real-time reenactment of human portrait videos," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 164, 2018.

[18] D. H. Brainard and B. A. Wandell, "Analysis of the retinex theory of color vision," *JOSA A*, vol. 3, no. 10, pp. 1651–1661, 1986.

[19] A. Geitgey, "Face recognition." [Online]. Available: https://github.com/ageitgey/face_recognition

[20] Wikipedia, "Savitzky–golay filter." [Online]. Available: https://en.wikipedia.org/wiki/Savitzky%E2%80%93Golay_filter

[21] C. Zaiontz, "Basic concepts of correlation." [Online]. Available: http://www.real-statistics.com/correlation/basic-concepts-correlation/

[22] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: identifying density-based local outliers," in *ACM sigmod record*, vol. 29, no. 2. ACM, 2000, pp. 93–104.

[23] S. Tripathy, "Icface: Interpretable and controllable face reenactment using gans," 2019. [Online]. Available: https://github.com/Blade6570/icface

[24] S. Ren, X. Cao, Y. Wei, and J. Sun, "Face alignment at 3000 fps via regressing local binary features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1685–1692.

[25] P. Garrido, L. Valgaerts, O. Rehmsen, T. Thormahlen, P. Perez, and C. Theobalt, "Automatic face reenactment," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4217–4224.

[26] P. Garrido, L. Valgaerts, H. Sarmadi, I. Steiner, K. Varanasi, P. Perez, and C. Theobalt, "Vdub: Modifying face video of actors for plausible visual alignment to a dubbed audio track," in *Computer graphics forum*, vol. 34, no. 2. Wiley Online Library, 2015, pp. 193–204.

[27] M. Kawai, T. Iwao, D. Mima, A. Maejima, and S. Morishima, "Data-driven speech animation synthesis focusing on realistic inside of the mouth," *Journal of information processing*, vol. 22, no. 2, pp. 401–409, 2014.

[28] S. Tripathy, J. Kannala, and E. Rahtu, "Icface: Interpretable and controllable face reenactment using gans," *arXiv preprint arXiv:1904.01909*, 2019.

[29] O. Wiles, A. Sophia Koepke, and A. Zisserman, "X2face: A network for controlling face generation using images, audio, and pose codes," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 670–686.

[30] M. Zollhöfer, J. Thies, P. Garrido, D. Bradley, T. Beeler, P. Pérez, M. Stamminger, M. Nießner, and C. Theobalt, "State of the art on monocular 3d face reconstruction, tracking, and applications," in *Computer Graphics Forum*, vol. 37, no. 2. Wiley Online Library, 2018, pp. 523–550.

[31] J. Thies, M. Zollhöfer, and M. Nießner, "Deferred neural rendering: Image synthesis using neural textures," *arXiv preprint arXiv:1904.12356*, 2019.

[32] H. Kim, P. Carrido, A. Tewari, W. Xu, J. Thies, M. Niessner, P. Pérez, C. Richardt, M. Zollhöfer, and C. Theobalt, "Deep video portraits," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 4, p. 163, 2018.

[33] M. Elgharib, M. BR, A. Tewari, H. Kim, W. Liu, H.-P. Seidel, and C. Theobalt, "Egoface: Egocentric face performance capture and videorealistic reenactment," *arXiv preprint arXiv:1905.10822*, 2019.

[34] D. Cozzolino, J. Thies, A. Rössler, C. Riess, M. Nießner, and L. Verdoliva, "Forensictransfer: Weakly-supervised domain adaptation for forgery detection," *arXiv preprint arXiv:1812.02510*, 2018.

[35] T. J. De Carvalho, C. Riess, E. Angelopoulou, H. Pedrini, and A. de Rezende Rocha, "Exposing digital image forgeries by illumination color classification," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 7, pp. 1182–1194, 2013.