FACE ANTI-SPOOFING: DETECTION, GENERALIZATION, AND VISUALIZATION

By

Yaojie Liu

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science - Doctor of Philosophy

2021

ABSTRACT

FACE ANTI-SPOOFING: DETECTION, GENERALIZATION, AND VISUALIZATION

By

Yaojie Liu

Face anti-spoofing is the process of distinguishing genuine faces and face presentation attacks: attackers presenting spoofing faces (e.g. photograph, digital screen, and mask) to the face recognition system and attempting to be authenticated as the genuine user. In recent years, face anti-spoofing has brought increasing attention to the vision community as it is a crucial step to prevent face recognition systems from a security breach. Previous approaches formulate face anti-spoofing as a binary classification problem, and many of them struggle to generalize to different conditions (such as pose, lighting, expressions, camera sensors and unknown spoof types). Moreover, those methods work as a black-box and cannot provide interpretation or visualization to their decision. To address those challenges, we investigate face anti-spoofing in 3 stages: detection, generalization and visualization. In the detection stage, we learn a CNN-RNN model to estimate auxiliary tasks of face depth and rPPG signals estimation, which can bring additional knowledge for the spoof detection. In the generalization stage, we investigate the detection of unknown spoof attacks and propose a novel Deep Tree Network (DTN) to well represent the unknown spoof attacks. In the visualization stage, we find "spoof trace, the subtle image pattern in spoof faces (e.g., color distortion, 3D mask edge, and Moire pattern), is effective to explain why a spoof is a spoof. We provide a proper physical modeling of the spoof traces and design a generative model to disentangle the spoof traces from input faces. In addition, we also show that a proper physical modeling can benefit other face problems, such as face shadow detection and removal. A proper shadow modeling can not only detect the shadow region effectively, but also remove the shadow in a visually plausible manner.

ACKNOWLEDGMENTS

Throughout preparing this dissertation I have received a great deal of support and assistance.

First of all, I would like to express my sincere gratitude to my advisor, Prof. Xiaoming Liu for his invaluable advice, continuous support, and patience during my PhD study. Your knowledge, experience, and enthusiasm motivate me to improve my academic research as well as life planning. I would like to thank Prof. Arun Ross for his lead on the anti-spoofing project. Your lead provides me with clear directions and great support of conducting research and achieving project goals. I would also like to thank the rest of my thesis committee: Prof. Anil Jain and Prof. Daniel Morris, for your insightful comments and suggestions, which push me to work on a boarder research impact. It's my honor to have you as my committee.

My sincere thanks also goes to all my co-authors, Dr. Amin Jourabloo, Dr. Yousef Atoum, Joel Stehouwer, and Xiaohong Liu, for working with me on all these exciting research projects and hold on tight in catching those deadlines. I also would like to thank my mentors, Dr. Barry Theobald and Dr. Nicholas Apostolof from Apple, and Dr. Xinyu Huang and Dr. Liu Ren from Bosch, for offering me the summer internship opportunities and leading me working on diverse exciting projects. Special shout-out to Christopher Perry. I really enjoy working with you on the anti-spoofing project and appreciate your skills and help on the face PAD solutions.

I thank my fellow labmates in Computer Vision Labs: Joseph Roth, Xi Yin, Luan Tran, Ying Tai, Bangjie Yin, Ziyuan Zhang, Garrick Brazil, Feng Liu, Shengjie Zhu, Masa Hu, Andrew Hou, Abhinav Kumar, Vishal Asnani, and Xiao Guo, for sharing and exchanging knowledge and opinions, and for all the fun we have had in all the activities. Also I thank my friends: Jialin Liu, Zhiwei Wang, Joshua Englesma, Xue Jiang, He Zhang, Jia Xue, and the list goes on and on.

Special thanks to my girlfriend Yufeng Wang and my dog Bagel the beagle, for enlightening my

life and giving me emotional support.

Last but not the least, I would like to thank my parents for your unconditional love and sacrifices. Thousands of words are not even enough to express my gratitude and love to you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

”For interpretation of the references to color in this and all other figures, the reader is referred to the electronic version of this thesis.”

Chapter 1

Introduction to Face Anti-Spoofing

1.1 Introduction

Biometrics utilize physiological, such as fingerprint, face, and iris, or behavioral characteristics, such as typing rhythm and gait, to uniquely identify or authenticate an individual. As biometric systems are widely used in real-world applications including mobile phone authentication and access control, biometric spoof, or Presentation Attack (PA) are becoming a large threat, where a spoofed biometric sample is presented to the biometric system and attempted to be authenticated. Face, as one of the most popular modalities, has received increasing attention in the academia and industry in the recent years (e.g., iPhone X). However, the attention also brings a growing incentive for hackers to design biometric presentation attacks (PA), or spoofs, to be authenticated as the genuine user. Due to the almost no-cost access to the human face, the spoof face can be as simple as a printed photo paper (i.e., print attack) and a digital image/video (i.e., replay attack), or as complicated as a 3D Mask and facial cosmetic makeup. With proper handling, those spoofs can be visually very close to the genuine users live face. As a result, these call for the need of developing robust face anti-spoofing algorithms.

In order to develop a face recognition system that is invulnerable to various types of PAs, there is an increasing demand on designing a robust face anti-spoofing (or PA detection) system to classify a face sample as live or spoof before recognizing its identity. As RGB image and video are the standard input to face recognition systems, most face anti-spoofing studies are RGB-based, either

single image or a clip of video. Previous approaches to tackle face anti-spoofing can be categorized in three groups. The first is the motion-based methods that aim at classifying face videos based on detecting movements of facial parts. Eye-blinking is one cue proposed in Pan et al. (2007); Sun et al. (2007), to detect spoof attacks such as paper attack. In Kollreider et al. (2007), Kollreider et al. use lip motion to monitor the face liveness. Methods proposed in Chetty (2010); Chetty & Wagner (2006) combine audio and visual cues to verify the face liveness. These methods are suitable for static attacks, but not dynamic attacks such as replay or mask attacks. The second is image quality and reflectance-based methods, which design features to capture the superimposed illumination and noise information to the spoof images. As the image quality factors are heuristic based on human observation (not data-driven), they show very limited capability to generalize to complex situation with variations such as lighting, pose, cameras and expressions. The third is the texture-based methods, which discover discriminative texture characteristics unique to various attack mediums. Compared to the previous two groups, texture-based methods explore the intrinsic properties of spoofing material and medium, and thus is more generalizable. However, due to a lack of understanding between pixel intensities and different types of attacks in the early studies, extracting robust texture features was challenging.

Most texture-based works utilize hand-crafted features and adopts shallow learning techniques (e.g., SVM and LDA) to develop an anti-spoofing system. Common local features that have been used in prior work include LBP in de Freitas Pereira et al. (2012, 2013); Määttä et al. (2011), HOG in Komulainen et al. (2013a); Yang et al. (2013), DoG in Peixoto et al. (2011); Tan et al. (2010), SIFT in Patel et al. (2016b) and SURF in Boulkenafet et al. (2017a). However, the aforementioned features to detect texture difference could be very sensitive to different illuminations, camera devices and specific identities. Researchers also seek solutions on different color spaces such as HSV and YCbCrin Boulkenafet et al. (2015, 2016), Fourier spectra Li et al. (2004) and Optical Flow Maps

(OFM)in Bao et al. (2009).

With CNN proven to successfully outperform other learning paradigms in many computer vision tasks in Kalchbrenner et al. (2014); Krizhevsky et al. (2012); Lawrence et al. (1997), it is then introduced as a new approach to handle face anti-spoofing. In Li et al. (2016a); Patel et al. (2016a), the CNN serves as a feature extractor. Both methods fine-tune their network from a pretrained model (CaffeNet in Patel et al. (2016a), VGG-face model in inLi et al. (2016a)), and extract the features to distinguish live vs. spoof. Yang et al. (2014) propose to learn a CNN as a classifier for face anti-spoofing. Registered face images with different spatial scales are stacked as input and live/spoof labeling is assigned as the output. In addition, Feng et al. Feng et al. (2016) propose to use multiple cues as the CNN input for live/spoof classification. They select Shearlet-based features to measure the image quality and the OFM of the face area as well as the whole scene area. And in Xu et al. (2015), Xu et al. propose an LSTM-CNN architecture to conduct a joint prediction for multiple frames of a video.

Though CNN-based methods provide significant improvement in terms of detection accuracy to face anti-spoofing, compared to other face related problems such as face recognition and face alignment, there are still substantially less efforts and exploration on face anti-spoofing using deep learning techniques. Therefore, in this work we aim to further explore the capability of CNN in handling face anti-spoofing, mainly in three aspects: improving the detection performance, generalization toward different domains such as unseen/unknown spoof types and capturing camera sensors, and providing visualization to the CNN's prediction.

1.2 Overview of the thesis

In Chapter 2, we argue the importance of auxiliary supervision to guide the learning toward more discriminative cues. A CNN-RNN model is learned to estimate the face depth with pixel-wise supervision, and to estimate rPPG signals with sequence-wise supervision. The estimated depth and rPPG are fused to distinguish live vs. spoof faces. Further, we introduce a new face anti-spoofing database that covers a large range of illumination, subject, and pose variations. Experiments show that our model achieves the state-of-theart results on both intra- and cross-database testing.

While advanced face anti-spoofing methods are developed, new types of spoof attacks are also being created and becoming a threat to all existing systems. To study the generalization of the face anti-spoofing methods, in Chapter 3, we define the detection of unknown spoof attacks as Zero-Shot Face Anti-spoofing (ZSFA). Previous ZSFA works only study 1-2 types of spoof attacks, such as print/replay, which limits the insight of this problem. In this chapter, we investigate the ZSFA problem in a wide range of 13 types of spoof attacks, including print, replay, 3D mask, and so on. A novel Deep Tree Network (DTN) is proposed to partition the spoof samples into semantic sub-groups in an unsupervised fashion. When a data sample arrives, being know or unknown attacks, DTN routes it to the most similar spoof cluster, and makes the binary decision. In addition, to enable the study of ZSFA, we introduce the first face anti-spoofing database that contains diverse types of spoof attacks. Experiments show that our proposed method achieves the state of the art on multiple testing protocols of ZSFA.

To gain a better visual understanding and interpretation of the spoof attacks, in Chapter 4, we identify a new problem of spoof trace disentangling. We show that the key to face anti-spoofing lies in the subtle image pattern, termed "spoof trace", *e.g.*, color distortion, 3D mask edge, Moir pattern, and many others. Spoof trace disentangling is motivated by the noise modeling and denoising

algorithms, for the purpose of anti-spoofing: inversely decomposing a spoof face into a spoof trace and a live face, and then utilizing the spoof trace for classification. Designing a generic anti-spoofing model to estimate those spoof traces can improve not only the generalization of the spoof detection, but also the interpretability of the model's decision. We first provide a proper modeling of spoof trace as an additive noise to the genuine face. We designs a novel adversarial learning framework to disentangle the spoof traces from input faces as a hierarchical combination of patterns at multiple scales. With the disentangled spoof traces, we unveil the live counterpart of the original spoof face, and further synthesize realistic new spoof faces after a proper geometric correction. Our method demonstrates superior spoof detection performance on both seen and unseen spoof scenarios while providing visually-convincing estimation of spoof traces.

In Chapter 5, we show that a proper physical modeling can benefit face shadow detection and removal problem as well. In-the-wild face photographs often suffer from undesired foreign shadows cast by external objects, e.g., hands, phones,and trees. Removing facial foreign shadows not only improves image aesthetics but also mitigates the negative impacts on face-related tasks. We tackle the blind removal of facial foreign shadow for both single image and videos, by making three contributions. Firstly, we propose a novel two-stage shadow modeling that consists of gray-scale shadow removal and colorization. This modeling provides an effective way to handle both color distortion and subsurface scattering effects. Second, we propose a novel Temporal Sharing Module to extract hierarchical features across multiple aligned video frames, which represents the shadow-free faces. Third, we collect a real face database with 280 videos captured under highly dynamic environments and annotate pixel-level shadow segmentation maps. Extensive experiments demonstrate the effectiveness of our approach com-paring with both the baseline and state-of-the-art methods

To better understand the work in this dissertation, Tab. 1.1 lists all the terms and the definitions

Terms	Definition
Live face	Genuine face from the subject without any physical manipulation of its identity. Also known as **bona fide** face.
Spoof face	Face not from the original subject. Also known as **presentation attack**.
Impersonation attack	Attacks in which the attacker wants to be recognized as a different subject.
Obfuscation attack	Attacks in which the attacker want to hide the identity of the attacker.
Spoof medium	Material used to present the spoof face, such as printed photo and digital screen.
Spoof trace	Patterns only existing in the spoof faces, such as moire pattern, and 3D mask edges.

Table 1.1 The term definition used in this work.

used in this work.

1.2.1 Contributions of the thesis

In this section, we list the contributions in this dissertation:

◇ Previous work regard the face anti-spoofing as a binary classification. We propose to leverage novel auxiliary information (i.e., depth map and rPPG) to supervise the CNN-RNN learning for improved detection performance. The auxiliary supervisions bring additional knowledge to benefit the face anti-spoofing;

◇ We conduct an extensive study of the generalization problem of face anti-spoofing. Specially, we study zero-shot face anti-spoofing on 13 different types of spoof attacks and propose a Deep Tree Network (DTN) to learn features hierarchically. DTN leverages existing spoof attack knowledge to effectively represent the unknown spoof attacks;

◇ To provide visual interpretation, we study a novel problem of spoof trace disentangling and propose a novel modeling to disentangle spoof traces into a hierarchical representation on generic spoof attacks. To our knowledge, these are the first work to solve face anti-spoofing in a generative and visually-intuitive approach;

◇ Inspired by the effective spoof trace modeling, we provide a effective modeling on the face with foriegn shadow, and propose a novel approach to decompose RGB shadow removal into grayscale shadow removal and colorization, and a temporal sharing module to ensure video consistency;

◇ We collect databases for face anti-spoofing, including SiW and SiW-M, and for face shadow detection, termed SFW.

Chapter 2

Detection: Face Anti-Spoofing with Auxiliary Supervisions

2.1 Introduction

RGB image and video are the standard input to face anti-spoofing systems, similar to face recognition systems. Researchers start the texture-based anti-spoofing approaches by feeding handcrafted features to binary classifiers Boulkenafet et al. (2017a); de Freitas Pereira et al. (2012, 2013); Komulainen et al. (2013a); Määttä et al. (2011); Mirjalili & Ross (2017); Patel et al. (2016b); Yang et al. (2013). Later in the deep learning era, several Convolutional Neural Networks (CNN) approaches utilize softmax loss as the supervision Feng et al. (2016); Li et al. (2016a); Patel et al. (2016a); Yang et al. (2014). It appears almost all prior work regard the face anti-spoofing problem as merely a *binary* (live vs. spoof) classification problem.

There are two main issues in learning deep anti-spoofing models with binary supervision, shown in Fig. 2.1. First, there are different levels of image degradation, namely *spoof patterns*, comparing a spoof face to a live one, which consist of skin detail loss, color distortion, moiré pattern, shape deformation and spoof artifacts (e.g., reflection) Li et al. (2004); Patel et al. (2016b). A CNN with softmax loss might discover *arbitrary* cues that are able to separate the two classes, such as screen bezel, but not the *faithful* spoof patterns. When those cues disappear during testing, these models would fail to distinguish spoof vs. live faces and result in poor generalization. Second, during

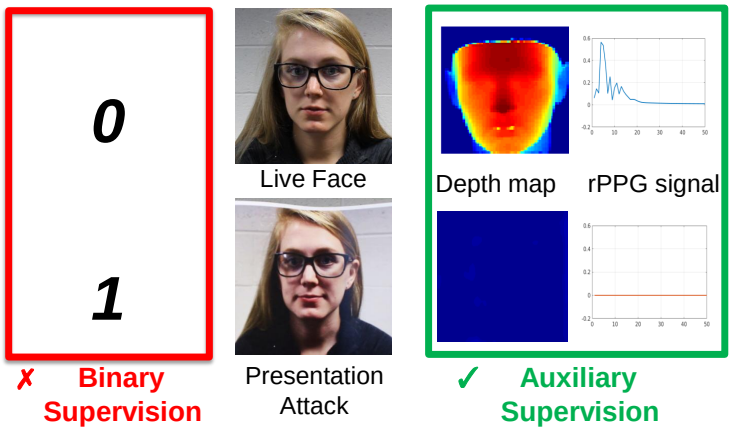


Figure 2.1 Conventional CNN-based face anti-spoof approaches utilize the binary supervision, which may lead to overfitting given the enormous solution space of CNN. This work designs a novel network architecture to leverage two auxiliary information as supervision: the depth map and rPPG signal, with the goals of improved generalization and explainable decisions during inference.

the testing, models learnt with binary supervision will only generate a binary decision without *explanation* or *rationale* for the decision. In the pursuit of Explainable Artificial Intelligence Turek (2016), it is desirable for the learnt model to generate the spoof patterns that support the final binary decision.

To address these issues, as shown in Fig. 2.2, we propose a deep model that uses the supervision from both the *spatial* and *temporal auxiliary information* rather than binary supervision, for the purpose of robustly detecting face PA from a face video. These auxiliary information are acquired based on our domain knowledge about the key *differences* between live and spoof faces, which include two perspectives: spatial and temporal. From the spatial perspective, it is known that live faces have face-like depth, e.g., the nose is closer to the camera than the cheek in frontal-view faces, while faces in print or replay attacks have flat or planar depth, e.g., all pixels on the image of a paper have the same depth to the camera. Hence, depth can be utilized as auxiliary information to supervise both live and spoof faces. From the temporal perspective, it was shown that the normal rPPG signals (i.e., heart pulse signal) are detectable from live, but not spoof, face videos Liu

et al. (2016b); Nowara et al. (2017). Therefore, we provide different rPPG signals as auxiliary supervision, which guides the network to learn from live or spoof face videos respectively. To enable both supervisions, we design a network architecture with a short-cut connection to capture different scales and a novel non-rigid registration layer to handle the motion and pose change for rPPG estimation.

Furthermore, similar to many vision problems, data plays a significant role in training the anti-spoofing models. As we know, camera/screen quality is a critical factor to the quality of spoof faces. Existing face anti-spoofing databases, such as NUAA Tan et al. (2010), CASIA Zhang et al. (2012), Replay-Attack Chingovska et al. (2012), and MSU-MFSD Wen et al. (2015), were collected $3-5$ years ago. Given the fast advance of consumer electronics, the types of equipment (e.g., cameras and spoofing mediums) used in those data collection are outdated compared to the ones nowadays, regarding the resolution and imaging quality. More recent MSU-USSA Patel et al. (2016b) and OULU databases Boulkenafet et al. (2017b) have subjects with fewer variations in poses, illuminations, expressions (PIE). The lack of necessary variations would make it hard to learn an effective model. Given the clear need for more advanced databases, we collect a face anti-spoofing database, named Spoof in the Wild Database (SiW). SiW database consists of 165 subjects, 6 spoofing mediums, and 4 sessions covering variations such as PIE, distance-to-camera, *etc*. SiW covers much larger variations than previous databases, as detailed in Tab. 2.1 and Sec. 2.4. The main contributions of this work include:

$\diamond$ We propose to leverage novel auxiliary information (i.e., depth map and rPPG) to supervise the CNN learning for improved generalization.

$\diamond$ We propose a novel CNN-RNN architecture for end-to-end learning the depth map and rPPG signal.

$\diamond$ We release a new database that contains variations of PIE, and other practical factors. We

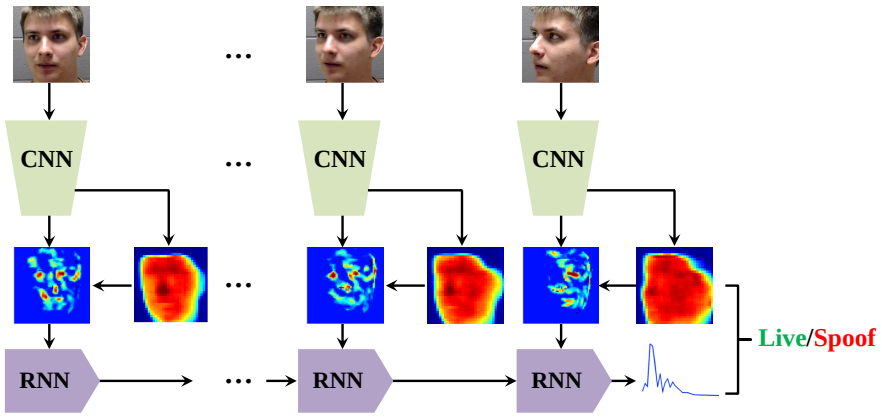


Figure 2.2 The overview of the proposed method.

achieve the state-of-the-art performance for face anti-spoofing.

2.2 Prior Work

We review the prior face anti-spoofing works in three groups: texture-based methods, temporal-based methods, and remote photoplethysmography methods.

Texture-based Methods Since most face recognition systems adopt only RGB cameras, using texture information has been a natural approach to tackling face anti-spoofing. Many prior works utilize hand-crafted features, such as LBP de Freitas Pereira et al. (2012, 2013); Määttä et al. (2011), HoG Komulainen et al. (2013a); Yang et al. (2013), SIFT Patel et al. (2016b) and SURF Boulkenafet et al. (2017a), and adopt traditional classifiers such as SVM and LDA. To overcome the influence of illumination variation, they seek solutions in a different input domain, such as HSV and YCbCr color space Boulkenafet et al. (2015, 2016), and Fourier spectrum Li et al. (2004).

As deep learning has proven to be effective in many computer vision problems, there are many recent attempts of using CNN-based features or CNNs in face anti-spoofing Feng et al. (2016); Li et al. (2016a); Patel et al. (2016a); Yang et al. (2014). Most of the work treats face anti-spoofing as a simple *binary* classification problem by applying the softmax loss. For example, Li et al. (2016a);

Patel et al. (2016a) use CNN as feature extractor and fine-tune from ImageNet-pretrained CaffeNet and VGG-face. The work of Feng et al. (2016); Li et al. (2016a) feed different designs of the face images into CNN, such as multi-scale faces and hand-crafted features, and directly classify live vs. spoof. One prior work that shares the similarity with ours is Atoum et al. (2017), where Atoum *et. al.* propose a two-steam CNN-based anti-spoofing method using texture and depth. We advance Atoum et al. (2017) in a number of aspects, including fusion with temporal supervision (i.e., rPPG), finer architecture design, novel non-rigid registration layer, and comprehensive experimental support.

Temporal-based Methods One of the earliest solutions for face anti-spoofing is based on temporal cues such as eye-blinking Pan et al. (2007); Patel et al. (2016a). Methods such as Kollreider et al. (2007); Shao et al. (2017) track the motion of mouth and lip to detect the face liveness. While these methods are effective to typical paper attacks, they become vulnerable when attackers present a replay attack or a paper attack with eye/mouth portion being cut.

There are also methods relying on more general temporal features, instead of the specific facial motion. The most common approach is frame concatenation. Many handcrafted feature-based methods may improve intra-database testing performance by simply concatenating the features of consecutive frames to train the classifiers Boulkenafet et al. (2015); de Freitas Pereira et al. (2012); Komulainen et al. (2013b). Additionally, there are some works proposing temporal-specific features, e.g., Haralick features Agarwal et al. (2016), motion mag Bharadwaj et al. (2014), and optical flow Bao et al. (2009). In the deep learning era, Feng *et. al.* feed the optical flow map and Shearlet image feature to CNN Feng et al. (2016). In Xu et al. (2015), Xu *et. al.* propose an LSTM-CNN architecture to utilize temporal information for binary classification. Overall, all prior methods still regard face anti-spoofing as a binary classification problem, and thus they have a hard time to generalize well in the cross-database testing. In this work, we extract discriminative temporal

information by learning the rPPG signal of the face video.

Remote Photoplethysmography (rPPG) Remote photoplethysmography (rPPG) is the technique to track vital signals, such as heart rate, without any contact with human skin Bobbia et al. (2016); de Haan & Jeanne (2013); Po et al. (2017); Tulyakov et al. (2016); Wu et al. (2016). Research starts with face videos with no motion or illumination change to videos with multiple variations. In de Haan & Jeanne (2013), Haan *et. al.* estimate rPPG signals from RGB face videos with lighting and motion changes. It utilizes color difference to eliminate the specular reflection and estimate two orthogonal chrominance signals. After applying the Band Pass Filter (BPM), the ratio of the chrominance signals are used to compute the rPPG signal.

rPPG has previously been utilized to tackle face anti-spoofing Liu et al. (2016b); Nowara et al. (2017). In Liu et al. (2016b), rPPG signals are used for detecting the 3D mask attack, where the live faces exhibit a pulse of heart rate unlike the 3D masks. They use rPPG signals extracted by de Haan & Jeanne (2013) and compute the correlation features for classification. Similarly, Magdalena *et. al.* Nowara et al. (2017) extract rPPG signals (also via de Haan & Jeanne (2013)) from three face regions and two non-face regions, for detecting print and replay attacks. Although in replay attacks, the rPPG extractor might still capture the normal pulse, the combination of multiple regions can differentiate live vs. spoof faces. While the analytic solution to rPPG extraction de Haan & Jeanne (2013) is easy to implement, we observe that it is sensitive to PIE variations. Hence, we employ a novel CNN-RNN architecture to *learn* a mapping from a face video to the rPPG signal, which is not only robust to PIE variations, but also discriminative to live *vs.* spoof.

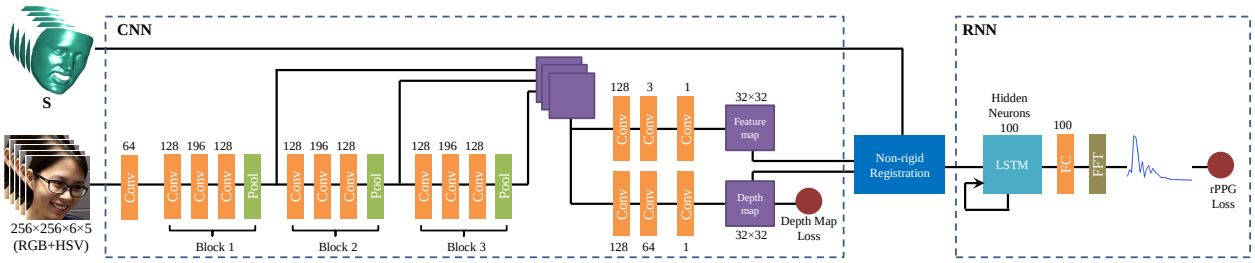


Figure 2.3 The proposed CNN-RNN architecture. The number of filters are shown on top of each layer, the size of all filters is 3×3 with stride 1 for convolutional and 2 for pooling layers. *Color code* used: *orange*=convolution, *green*=pooling, *purple*=response map.

2.3 Face Anti-Spoofing with Deep Network

The main idea of the proposed approach is to guide the deep network to focus on the *known spoof patterns* across spatial and temporal domains, rather than to extract any cues that could separate two classes but are not generalizable. As shown in Fig. 2.2, the proposed network combines CNN and RNN architectures in a coherent way. The CNN part utilizes the depth map supervision to discover subtle texture property that leads to distinct depths for live and spoof faces. Then, it feeds the estimated depth and the feature maps to a novel *non-rigid registration* layer to create aligned feature maps. The RNN part is trained with the aligned maps and the rPPG supervision, which examines temporal variability across video frames.

2.3.1 Depth Map Supervision

Depth maps are a representation of the 3D shape of the face in a 2D image, which shows the face location and the depth information of different facial areas. This representation is more informative than binary labels since it indicates one of the fundamental differences between live faces, and print and replay PA. We utilize the depth maps in the depth loss function to supervise the CNN part. The pixel-based depth loss guides the CNN to learn a mapping from the face area within a receptive field to a labeled depth value – a scale within $[0, 1]$ for live faces and 0 for spoof faces.

To estimate the depth map for a 2D face image, given a face image, we utilize the state-of-the-art dense face alignment (DeFA) methods Jourabloo & Liu (2017); Liu et al. (2017) to estimate the 3D shape of the face. The frontal dense 3D shape $\mathbf{S}_F \in \mathbb{R}^{3\times Q}$, with Q vertices, is represented as a linear combination of identity bases $\{\mathbf{S}_{id}^i\}_{i=1}^{N_{id}}$ and expression bases $\{\mathbf{S}_{exp}^i\}_{i=1}^{N_{exp}}$,

$$\mathbf{S}_F = \mathbf{S}_0 + \sum_{i=1}^{N_{id}} \alpha_{id}^i \mathbf{S}_{id}^i + \sum_{i=1}^{N_{exp}} \alpha_{exp}^i \mathbf{S}_{exp}^i, \tag{2.1}$$

where $\alpha_{id} \in \mathbb{R}^{199}$ and $\alpha_{ext} \in \mathbb{R}^{29}$ are the identity and expression parameters, and $\alpha = [\alpha_{id}, \alpha_{exp}]$ are the shape parameters. We utilize the Basel 3D face model Paysan et al. (2009) and the facewearhouse Cao et al. (2014) as the identity and expression bases.

With the estimated pose parameters $\mathbf{P} = (s, \mathbf{R}, \mathbf{t})$, where $\mathbf{R}$ is a 3D rotation matrix, $\mathbf{t}$ is a 3D translation, and s is a scale, we align the 3D shape $\mathbf{S}$ to the 2D face image:

$$\mathbf{S} = s\mathbf{R}\mathbf{S}_F + \mathbf{t}. \tag{2.2}$$

Given the challenge of estimating the *absolute* depth from a 2D face, we normalize the z values of 3D vertices in $\mathbf{S}$ to be within $[0, 1]$. That is, the vertex closest to the camera (e.g., nose) has a depth of one, and the vertex furthest away has the depth of zero. Then, we apply the Z-Buffer algorithm Zhu et al. (2016) to $\mathbf{S}$ for projecting the normalized z values to a 2D plane, which results in an estimated "ground truth" 2D depth map $\mathbf{D} \in \mathbb{R}^{32\times 32}$ for a face image.

2.3.2 rPPG Supervision

rPPG signals have recently been utilized for face anti-spoofing Liu et al. (2016b); Nowara et al. (2017). The rPPG signal provides temporal information about face liveness, as it is related to the

intensity changes of facial skin over time. These intensity changes are highly correlated with the blood flow. The traditional method de Haan & Jeanne (2013) for extracting rPPG signals has three drawbacks. First, it is sensitive to pose and expression variation, as it becomes harder to *track* a specific face area for measuring intensity changes. Second, it is also sensitive to illumination changes, since the extra lighting affects the amount of reflected light from the skin. Third, for the purpose of anti-spoof, rPPG signals extracted from spoof videos might not be sufficiently *distinguishable* to signals of live videos.

One novelty aspect of our approach is that, instead of computing the rPPG signal via de Haan & Jeanne (2013), our RNN part learns to estimate the rPPG signal. This eases the signal estimation from face videos with PIE variations, and also leads to more discriminative rPPG signals, as different rPPG supervisions are provided to live vs. spoof videos. We assume that the videos of the same subject under different PIE conditions have the *same* ground truth rPPG signal. This assumption is valid since the heart beat is similar for the videos of the same subject that are captured in a short span of time (< 5 minutes). The rPPG signal extracted from the constrained videos (i.e., no PIE variation) are used as the "ground truth" supervision in the rPPG loss function for *all* live videos of the same subject. This consistent supervision helps the CNN and RNN parts to be robust to the PIE changes.

In order to extract the rPPG signal from a face video without PIE, we apply the DeFA Liu et al. (2017) to each frame and estimate the dense 3D face shape. We utilize the estimated 3D shape to track a face region. For a tracked region, we compute two orthogonal chrominance signals $\mathbf{x}_f = 3\mathbf{r}_f - 2\mathbf{g}_f$, $\mathbf{y}_f = 1.5\mathbf{r}_f + \mathbf{g}_f - 1.5\mathbf{b}_f$ where $\mathbf{r}_f, \mathbf{g}_f, \mathbf{b}_f$ are the bandpass filtered versions of the $\mathbf{r}, \mathbf{g}, \mathbf{b}$ channels with the skin-tone normalization. We utilize the ratio of the standard deviation of the chrominance signals $\gamma = \frac{\sigma(\mathbf{x}_f)}{\sigma(\mathbf{y}_f)}$ to compute blood flow signals de Haan & Jeanne (2013). We

calculate the signal $\mathbf{p}$ as:

$$\mathbf{p} = 3(1 - \frac{\gamma}{2})\mathbf{r}_f - 2(1 + \frac{\gamma}{2})\mathbf{g}_f + \frac{3\gamma}{2}\mathbf{b}_f. \tag{2.3}$$

By applying FFT to $\mathbf{p}$, we obtain the rPPG signal $\mathbf{f} \in \mathbb{R}^{50}$, which shows the magnitude of each frequency.

2.3.3 Network Architecture

Our proposed network consists of two deep networks. First, a CNN part evaluates each frame separately and estimates the depth map and feature map of each frame. Second, a recurrent neural network (RNN) part evaluates the temporal variability across the feature maps of a sequence.

2.3.3.1 CNN Network

We design a Fully Convolutional Network (FCN) as our CNN part, as shown in Fig. 2.3. The CNN part contains multiple blocks of three convolutional layers, pooling and resizing layers where each convolutional layer is followed by one exponential linear layer and batch normalization layer. Then, the resizing layers resize the response maps after each block to a pre-defined size of 64×64 and concatenate the response maps. The bypass connections help the network to utilize extracted features from layers with different depths similar to the ResNet structure He et al. (2016). After that, our CNN has two branches, one for estimating the depth map and the other for estimating the feature map.

The first output of the CNN is the estimated depth map of the input frame $\mathbf{I} \in \mathbb{R}^{256\times256}$, which

is supervised by the estimated "ground truth" depth $\mathbf{D}$,

$$\Theta_D = \arg\min_{\Theta_D} \sum_{i=1}^{N_d} ||\text{CNN}_D(\mathbf{I}_i; \Theta_D) - \mathbf{D}_i||_1^2, \tag{2.4}$$

where Θ_D is the CNN parameters and N_d is the number of training images. The second output of the CNN is the feature map, which is fed into the non-rigid registration layer.

2.3.3.2 RNN Network

The RNN part aims to estimate the rPPG signal $\mathbf{f}$ of an input sequence with N_f frames $\{\mathbf{I}_j\}_{j=1}^{N_f}$. As shown in Fig. 2.3, we utilize one LSTM layer with 100 hidden neurons, one fully connected layer, and an FFT layer that converts the response of fully connected layer into the Fourier domain. Given the input sequence $\{\mathbf{I}_j\}_{j=1}^{N_f}$ and the "ground truth" rPPG signal $\mathbf{f}$, we train the RNN to minimize the ℓ_1 distance of the estimated rPPG signal to "ground truth" $\mathbf{f}$:

$$\Theta_R = \arg\min_{\Theta_R} \sum_{i=1}^{N_s} ||\text{RNN}_R([\{\mathbf{F}_j\}_{j=1}^{N_f}]_i; \Theta_R) - \mathbf{f}_i||_1^2, \tag{2.5}$$

where Θ_R is the RNN parameters, $\mathbf{F}_j \in \mathbb{R}^{32\times32}$ is the frontalized feature map (details in Sec. 2.3.4), and N_s is the number of sequences.

2.3.3.3 Implementation Details

Ground Truth Data Given a set of live and spoof face videos, we provide the ground truth supervision for the depth map $\mathbf{D}$ and rPPG signal $\mathbf{f}$, as in Fig. 2.4. We follow the procedure in Sec. 2.3.1 to compute "ground truth" data for live videos. For spoof videos, we set the ground truth depth maps to a plain surface, i.e., zero depth. Similarly, we follow the procedure in Sec. 2.3.2 to compute the "ground truth" rPPG signal from a patch on the forehead, for one live video of each

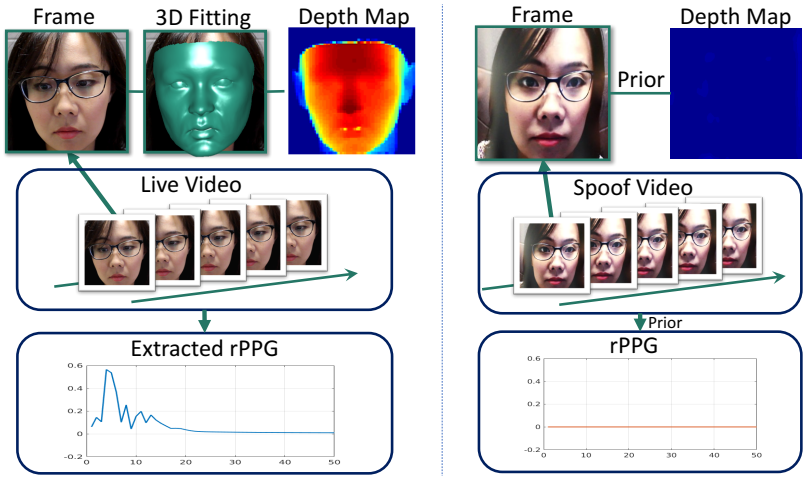


Figure 2.4 Example ground truth depth maps and rPPG signals.

subject without PIE variation. Also, we normalize the norm of estimated rPPG signal such that $\|\mathbf{f}\|_2 = 1$. For spoof videos, we consider the rPPG signals are zero.

Note that, while the term "depth" is used here, our estimated depth is different to the conventional depth map in computer vision. It can be viewed as a "pseudo-depth" and serves the purpose of providing discriminative auxiliary supervision to the learning process. The same perspective applies to the supervision based on pseudo-rPPG signal.

Training Strategy Our proposed network combines the CNN and RNN parts for end-to-end training. The desired training data for the CNN part should be from diverse subjects, so as to make the training procedure more stable and increase the generalizability of the learned model. Meanwhile, the training data for the RNN part should be long sequences to leverage the temporal information across frames. These two preferences can be contradictory to each other, especially given the limited GPU memory. Hence, to satisfy both preferences, we design a two-stream training strategy. The first stream satisfies the preference of the CNN part, where the input includes face images $\mathbf{I}$ and the ground truth depth maps $\mathbf{D}$. The second stream satisfies the RNN part, where the input includes face sequences $\{\mathbf{I}_j\}_{j=1}^{N_f}$, the ground truth depth maps $\{\mathbf{D}_j\}_{j=1}^{N_f}$, the estimated 3D shapes $\{\mathbf{S}_j\}_{j=1}^{N_f}$, and the corresponding ground truth rPPG signals $\mathbf{f}$. During training, our method

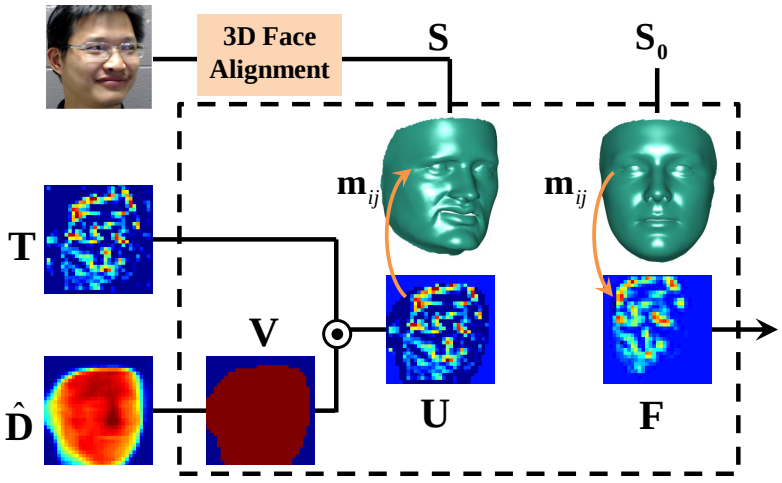


Figure 2.5 The non-rigid registration layer.

alternates between these two streams to converge to a model that minimizes both the depth map and rPPG losses. Note that even though the first stream only updates the weights of the CNN part, the back propagation of the second stream updates the weights of both CNN and RNN parts in an end-to-end manner.

Testing To provide a classification score, we feed the testing sequence to our network and compute the depth map $\hat{\mathbf{D}}$ of the last frame and the rPPG signal $\hat{\mathbf{f}}$. Instead of designing a classifier using $\hat{\mathbf{D}}$ and $\hat{\mathbf{f}}$, we compute the final score as:

$$score = ||\hat{\mathbf{f}}||_2^2 + \lambda||\hat{\mathbf{D}}||_2^2, \tag{2.6}$$

where λ is a constant weight for combining the two outputs of the network.

2.3.4 Non-rigid Registration Layer

We design a new non-rigid registration layer to prepare data for the RNN part. This layer utilizes the estimated dense 3D shape to align the activation or feature maps from the CNN part. This layer is important to ensure that the RNN tracks and learns the changes of the activation for the *same*

facial area across time, as well as across all subjects.

As shown in Fig. 2.5, this layer has three inputs: the feature map $\mathbf{T} \in \mathbb{R}^{32\times32}$, the depth map $\hat{\mathbf{D}}$ and the 3D shape $\mathbf{S}$. Within this layer, we first threshold the depth map and generate a binary mask $\mathbf{V} \in \mathbb{R}^{32\times32}$:

$$\mathbf{V} = \hat{\mathbf{D}} \geq threshold. \tag{2.7}$$

Then, we compute the inner product of the binary mask and the feature map $\mathbf{U} = \mathbf{T} \odot \mathbf{V}$, which essentially utilizes the depth map as a visibility indicator for each pixel in the feature map. If the depth value for one pixel is less than the threshold, we consider that pixel to be invisible. Finally, we frontalize $\mathbf{U}$ by utilizing the estimated 3D shape $\mathbf{S}$,

$$\mathbf{F}(i,j) = \mathbf{U}(\mathbf{S}(\mathbf{m}_{ij},1), \mathbf{S}(\mathbf{m}_{ij},2)), \tag{2.8}$$

where $\mathbf{m} \in \mathbb{R}^K$ is the pre-defined list of K indexes of the face area in $\mathbf{S_0}$, and $\mathbf{m}_{ij}$ is the corresponding index of pixel i,j. We utilize $\mathbf{m}$ to project the masked activation map $\mathbf{U}$ to the frontalized image $\mathbf{F}$. This proposed non-rigid registration layer has three contributions to our network:

$\diamond$ By applying the non-rigid registration, the input data are aligned and the RNN can compare the feature maps without concerning about the facial pose or expression. In other words, it can learn the temporal changes in the activation of the feature maps for the same facial area.

$\diamond$ The non-rigid registration removes the background area in the feature map. Hence the background area would not participate in RNN learning, although the background information is already utilized in the layers of the CNN part.

$\diamond$ For spoof faces, the depth maps are likely to be closer to zero. Hence, the inner product with the depth maps substantially weakens the activation in the feature maps, which makes it easier for

Dataset	# of subj.	# of sess.	# of live/attack video/image (I)	Pose range	Different expres.	Extra light.	Display devices	Spoof attacks
NUAA [Tan et al. (2010)]	15	3	5105/7509 (I)	Frontal	No	Yes	-	1 Print
CASIA-MFSD [Zhang et al. (2012)]	50	3	150/450 (V)	Frontal	No	No	iPad	1 Print, 1 Replay
Replay-Attack [Chingovska et al. (2012)]	50	1	200/1000 (V)	Frontal	No	Yes	iPhone 3GS, iPad	1 Print, 2 Replay
MSU-MFSD [Wen et al. (2015)]	35	1	110/330 (V)	Frontal	No	No	iPad Air, iPhone 5S	1 Print, 2 Replay
MSU-USSA [Patel et al. (2016b)]	1140	1	1140/9120 (I)	$< 45°$	Yes	Yes	MacBook, Nexus 5, Nvidia Shield Tablet	2 print, 6 Replay
Oulu-NPU [Boulkenafet et al. (2017b)]	55	3	1980/3960 (V)	Frontal	No	Yes	Dell 1905FP, Macbook Retina	2 Print, 2 Replay
SiW	165	4	1320/3300 (V)	$< 90°$	Yes	Yes	iPad Pro, iPhone 7, Galaxy S8, Asus MB168B	2 Print, 4 Replay

Table 2.1 The comparison of our collected SiW dataset with existing datasets for face anti-spoofing.

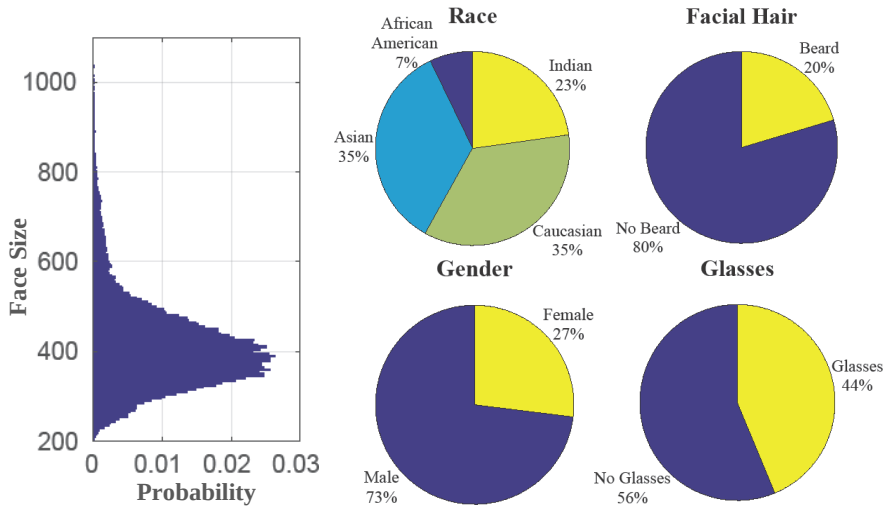


Figure 2.6 The statistics of the subjects in the SiW database. Left side: The histogram shows the distribution of the face sizes.

the RNN to output zero rPPG signals. Likewise, the back propagation from the rPPG loss also encourages the CNN part to generate zero depth maps for either all frames, or one pixel location in majority of the frames within an input sequence.

2.4 Collection of Face Anti-Spoofing Database

With the advance of sensor technology, existing anti-spoofing systems can be vulnerable to emerging high-quality spoof mediums. One way to make the system robust to these attacks is to collect new high-quality databases. In response to this need, we collect a new face anti-spoofing database named Spoof in the Wild (SiW) database, which has multiple advantages over previous datasets as

in Tab. 2.1. First, it contains substantially more live subjects with diverse races, e.g., 3 times of the subjects of Oulu-NPU. Note that MSU-USSA is constructed using existing images of celebrities without capturing live faces. Second, live videos are captured with two high-quality cameras (Canon EOS T6, Logitech C920 webcam) with different PIE variations.

SiW provides live and spoof 30-FPS videos from 165 subjects. For each subject, we have 8 live and 20 spoof videos, in total $4,620$ videos. Some statistics of the subjects are shown in Fig. 2.6. The live videos are collected in four sessions. In Session 1, the subject moves his head with varying distances to the camera. In Session 2, the subject changes the yaw angle of the head within $[-90^{\circ}, 90^{\circ}]$, and makes different face expressions. In Sessions $3, 4$, the subject repeats the Sessions $1, 2$, while the collector moves the point light source around the face from different orientations.

The live videos captured by both cameras are of $1,920 \times 1,080$ resolution. We provide two print and four replay video attacks for each subject, with examples shown in Fig. 2.7. To generate different qualities of print attacks, we capture a high-resolution image ($5,184 \times 3,456$) for each subject and use it to make a high-quality print attack. Also, we extract a frontal-view frame from a live video for lower-quality print attack. We print the images with an HP color LaserJet M652 printer. The print attack videos are captured by holding printed papers still or warping them in front of the cameras. To generate high-quality replay attack videos, we select four spoof mediums: Samsung Galaxy S8, iPhone 7, iPad Pro, and PC (Asus MB168B) screens. For each subject, we randomly select two of the four high-quality live videos to display in the spoof mediums.

Figure 2.7 Example live (top) and spoof (bottom) videos in SiW.

2.5 Experimental Results

2.5.1 Experimental Setup

Databases We evaluate our method on multiple databases to demonstrate its generalizability. We utilize SiW and Oulu databases Boulkenafet et al. (2017b) as new high-resolution databases and perform intra and cross testing between them. Also, we use the CASIA-MFSD Zhang et al. (2012) and Replay-Attack Chingovska et al. (2012) databases for cross testing and comparing with the state of the art.

Parameter setting The proposed method is implemented in TensorFlow Abadi et al. (2016) with a constant learning rate of $3\text{e}{-3}$, and 10 epochs of the training phase. The batch size of the CNN stream is 10 and that of the CNN-RNN stream is 2 with N_f being 5. We randomly initialize our network by using a normal distribution with zero mean and std of 0.02. We set λ in Eq. 4.10 to 0.015 and $threshold$ in Eq. 2.7 to 0.1.

Evaluation metrics To compare with prior works, we report our results with the following metrics: Attack Presentation Classification Error Rate $APCER$ ISO/IEC-JTC-1/SC-37 (2016), Bona Fide Presentation Classification Error Rate $BPCER$ ISO/IEC-JTC-1/SC-37 (2016), $ACER =$

FDR	1%	2%	10%	20%
Model 1	8.5%	18.1%	71.4%	81.0%
Model 2	40.2%	46.9%	78.5%	93.5%
Model 3	39.4%	42.9%	67.5%	87.5%
Model 4	**45.8%**	**47.9%**	**81%**	**94.2%**

Table 2.2 TDR at different FDRs, cross testing on Oulu Protocol 1.

Test \ Train	5	10	20
5	4.16%	4.16%	3.05%
10	4.02%	3.61%	2.78%
20	4.10%	3.67%	2.98%

Table 2.3 ACER of our method at different N_f, on Oulu Protocol 2.

$\frac{APCER+BPCER}{2}$ ISO/IEC-JTC-1/SC-37 (2016), and Half Total Error Rate $HTER$. The $HTER$ is half of the summation of the False Rejection Rate (FRR) and the False Acceptance Rate (FAR).

2.5.2 Experimental Comparison

2.5.2.1 Ablation Study

Advantage of proposed architecture We compare four architectures to demonstrate the advantages of the proposed loss layers and non-rigid registration layer. *Model* 1 has an architecture similar to the CNN part in our method (Fig. 2.3), except that it is extended with additional pooling layers, fully connected layers, and softmax loss for binary classification. *Model* 2 is the CNN part in our method with a depth map loss function. We simply use $||\hat{\mathbf{D}}||_2$ for classification. *Model* 3 contains the CNN and RNN parts without the non-rigid registration layer. Both of the depth map and rPPG loss functions are utilized in this model. However, the RNN part would process unregistered feature maps from the CNN. *Model* 4 is the proposed architecture.

We train all four models with the live and spoof videos from 20 subjects of SiW. We compute

the cross-testing performance of all models on Protocol 1 of Oulu database. The TDR at different FDR are reported in Tab. 2.2. *Model* 1 has a poor performance due to the binary supervision. In comparison, by only using the depth map as supervision, *Model* 2 achieves substantially better performance. However, after adding the RNN part with the rPPG supervision, our proposed *Model* 4 can further the performance improvement. By comparing *Model* 4 and 3, we can see the advantage of the non-rigid registration layer. It is clear that the RNN part cannot use feature maps directly for tracking the changes in the activations and estimating the rPPG signals.

Advantage of longer sequences To show the advantage of utilizing longer sequences for estimating the rPPG, we train and test our model when the sequence length N_f is 5, 10, or 20, using intra-testing on Oulu Protocol 2. From Tab. 2.3, we can see that by increasing the sequence length, the ACER decreases due to more reliable rPPG estimation. Despite the benefit of longer sequences, in practice, we are limited by the GPU memory size, and forced to decrease the image size to 128×128 for all experiments in Tab. 2.3. Hence, we set N_f to be 5 with the image size of 256×256 in subsequent experiments, due to importance of higher resolution (e.g, a lower *ACER* of 2.5% in Tab. 2.4 is achieved than 4.16%).

2.5.2.2 Intra Testing

We perform intra testing on Oulu and SiW databases. For Oulu, we follow the four protocols Boulkenafet (2017) and report their *APCER*, *BPCER* and *ACER*. Tab. 2.4 shows the comparison of our proposed method and the best two methods for *each* protocol respectively, in the face anti-spoofing competition Boulkenafet (2017). Our method achieves the lowest *ACER* in 3 out of 4 protocols. We have slightly worse *ACER* on Protocol 2. To set a baseline for future study on SiW, we define three protocols for SiW. The Protocol 1 deals with variations in face pose and expression. We train using the first 60 frames of the training videos that are mainly frontal view faces, and test on all

Prot.	Method	APCER (%)	BPCER (%)	ACER (%)
1	CPqD	2.9	10.8	6.9
	GRADIANT	**1.3**	12.5	6.9
	Ours	1.6	**1.6**	**1.6**
2	MixedFASNet	9.7	2.5	6.1
	Ours	**2.7**	2.7	2.7
	GRADIANT	3.1	**1.9**	**2.5**
3	MixedFASNet	5.3 ± 6.7	7.8 ± 5.5	6.5 ± 4.6
	GRADIANT	$\mathbf{2.6 \pm 3.9}$	5.0 ± 5.3	3.8 ± 2.4
	Ours	2.7 ± 1.3	$\mathbf{3.1 \pm 1.7}$	$\mathbf{2.9 \pm 1.5}$
4	Massy HNU	35.8 ± 35.3	$\mathbf{8.3 \pm 4.1}$	22.1 ± 17.6
	GRADIANT	$\mathbf{5.0 \pm 4.5}$	15.0 ± 7.1	10.0 ± 5.0
	Ours	9.3 ± 5.6	10.4 ± 6.0	$\mathbf{9.5 \pm 6.0}$

Table 2.4 The intra-testing results on four protocols of Oulu.

Prot.	Subset	Subject #	Attack	APCER (%)	BPCER (%)	ACER (%)
1	Train	90	First 60 Frames	3.58	3.58	3.58
	Test	75	All			
2	Train	90	3 display	0.57 ± 0.69	0.57 ± 0.69	0.57 ± 0.69
	Test	75	1 display			
3	Train	90	print (display)	8.31 ± 3.81	8.31 ± 3.80	8.31 ± 3.81
	Test	75	display (print)			

Table 2.5 The intra-testing results on three protocols of SiW.

testing videos. The Protocol 2 evaluates the performance of cross spoof medium of replay attack. The Protocol 3 evaluates the performance of cross PA, i.e., from print attack to replay attack and vice versa. Tab. 2.5 shows the protocol definition and our performance of each protocol.

2.5.2.3 Cross Testing

To demonstrate the generalization of our method, we perform multiple cross-testing experiments. Our model is trained with live and spoof videos of 80 subjects in SiW, and test on all protocols of Oulu. The *ACER* on Protocol 1-4 are respectively: 10.0%, 14.1%, $13.8 \pm 5.7\%$, and $10.0 \pm 8.8\%$. Comparing these cross-testing results to the *intra-testing* results in Boulkenafet (2017), we are ranked sixth on the average *ACER* of four protocols, among the 15 participants of the face anti-spoofing competition. Especially on Protocol 4, the hardest one among all protocols, we achieve the *same ACER* of 10.0% as the top performer. This is a notable result since cross testing is known to

Method	CASIA→Replay	Replay→CASIA
Motion [de Freitas Pereira et al. (2013)]	50.2%	47.9%
LBP [de Freitas Pereira et al. (2013)]	55.9%	57.6%
LBP-TOP [de Freitas Pereira et al. (2013)]	49.7%	60.6%
Motion-Mag [Bharadwaj et al. (2013)]	50.1%	47.0%
Spectral [Pinto et al. (2015)]	34.4%	50.0%
CNN [Yang et al. (2014)]	48.5%	45.5%
LBP [Boulkenafet et al. (2015)]	47.0%	39.6%
Colour Texture [Boulkenafet et al. (2016)]	30.3%	37.7%
Ours	**27.6%**	**28.4%**

Table 2.6 Cross testing on CASIA-MFSD vs. Replay-Attack.

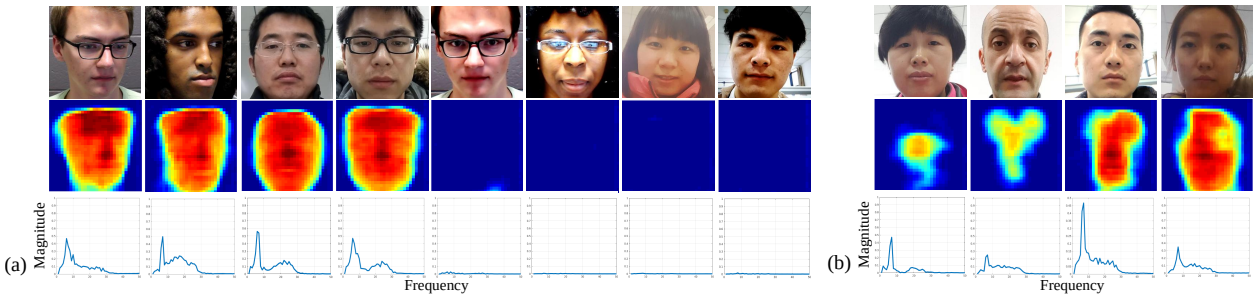


Figure 2.8 (a) 8 successful anti-spoofing examples and their estimated depth maps and rPPG signals. (b) 4 failure examples: the first two are live and the other two are spoof. Note our ability to estimate discriminative depth maps and rPPG signals.

be substantially harder than intra testing, and yet our cross-testing result is comparable with the top intra-testing performance. This demonstrates the generalization ability of our learnt model.

Furthermore, we utilize the CASIA-MFSD and Replay-Attack databases to perform cross testing between them, which is widely used as a cross-testing benchmark. Tab. 2.6 compares the cross-testing *HTER* of different methods. Our proposed method reduces the cross-testing errors on the Replay-Attack and CASIA-MFSD databases by 8.9% and 24.6% respectively, relative to the previous SOTA.

2.5.2.4 Visualization and Analysis

Examples of successful and failure cases in estimating depth maps and rPPG signals are shown in Fig. 2.8. In the proposed architecture, the frontalized feature maps are utilized as input to the

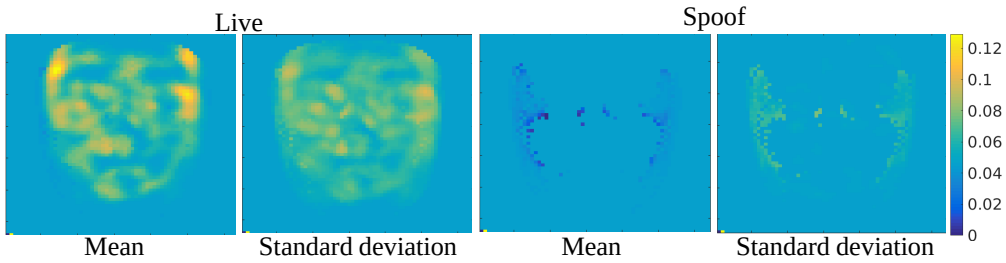


Figure 2.9 Mean/Std of frontalized feature maps for live and spoof.

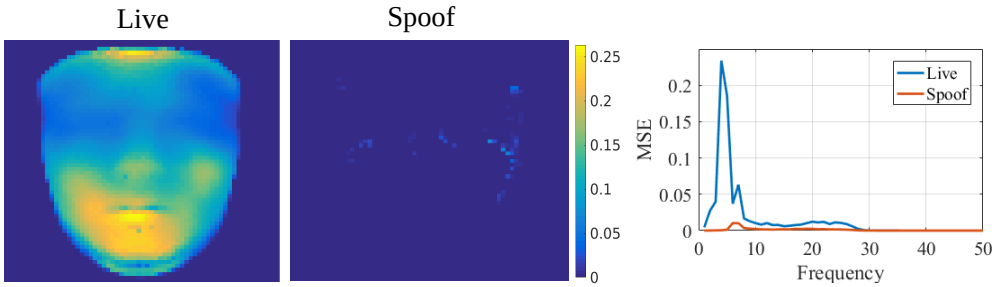


Figure 2.10 The MSE of estimating depth maps and rPPG signals.

RNN part and are supervised by the rPPG loss function. The values of these maps can show the importance of different facial areas to rPPG estimation. Fig. 2.9 shows the mean and standard deviation of frontalized feature maps, computed from 1,080 live and spoof videos of Oulu. We can see that the side areas of forehead and cheek have higher influence for rPPG estimation.

While the goal of our system is to detect PAs, our model is trained to estimate the auxiliary information. Hence, in addition to anti-spoof, we also like to evaluate the accuracy of auxiliary information estimation. For this purpose, we calculate the accuracy of estimating depth maps and rPPG signals, for testing data in Protocol 2 of Oulu. As shown in Fig. 2.10, the accuracy for both estimation in spoof data is high, while that of the live data is relatively lower. Note that the depth estimation of the mouth area has more errors, which is consistent with the fewer activations of the same area in Fig. 2.9.

Finally, we conduct statistical analysis on the failure cases, since our system can determine potential causes using the auxiliary information. With Proctocol 2 of Oulu, we identify 31 failure cases (2.7% *ACER*). For each case, we calculate whether anti-spoofing using its depth map or rPPG

signal would fail if that information alone is used. In total, $\frac{29}{31}$, $\frac{13}{31}$, and $\frac{11}{31}$ samples fail due to depth map, rPPG signals, or both. This indicates the future research direction.

2.6 Conclusions

This chapter identifies the importance of auxiliary supervision to deep model-based face anti-spoofing. The proposed network combines CNN and RNN architectures to jointly estimate the depth of face images and rPPG signal of face video. One significant improvement of auxiliary supervisions is to make each CNN prediction to be based on a local receipt field. It would effective reduce the CNN training from overfitting with limited data. We introduce the SiW database that contains more subjects and variations than prior databases. Finally, we experimentally demonstrate the superiority of our method.

Chapter 3

Generalization: Zero-shot and Open-set Face Anti-Spoofing

3.1 Introduction

Attackers can utilize a wide variety of mediums to launch spoof attacks. The most common ones are replaying videos/images on digital screens, i.e., replay attack, and printed photograph, i.e., print attack. Different methods are proposed to handle replay and print attacks, based on either handcrafted features Boulkenafet et al. (2015); Määttä et al. (2011); Patel et al. (2016b) or CNN-based features Atoum et al. (2017); Feng et al. (2016); Jourabloo et al. (2018); Liu et al. (2018c). Recently, high-quality 3D custom mask is also used for attacking, i.e., 3D mask attack. In Liu et al. (2018b, 2016a,b), methods for detecting print/replay attacks are found to be less effective for this new spoof, and hence the authors leverage the remote photoplethysmography (r-PPG) to detect the heart rate pulse as the spoofing cue. Further, facial makeup may also influence the outcome of recognition, i.e., makeup attack Chen et al. (2013). Many works Chang et al. (2018); Chen et al. (2013, 2014) study facial makeup, despite not as an anti-spoofing problem.

All aforementioned methods present algorithmic solutions to the *known* spoof attack(s), where models are trained and tested on the *same* type(s) of spoof attacks. However, in real-world applications, attackers can also initiate spoof attacks that we, the algorithm designers, are not aware

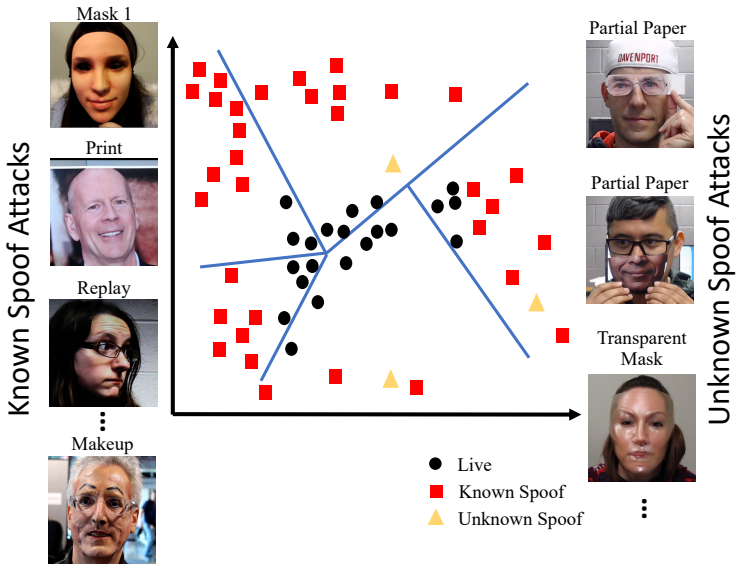


Figure 3.1 To detect unknown spoof attacks, we propose a Deep Tree Network (DTN) to unsupervisely learn a hierarchic embedding for known spoof attacks. Samples of unknown attacks will be routed through DTN and classified at the destined leaf node.

of, termed *unknown* spoof attacks[1]. Researchers increasingly pay attention to the generalization of anti-spoofing models, i.e., how well they are able to detect spoof attacks that have never been seen during the training? We define the problem of detecting unknown face spoof attacks as **Zero-Shot Face Anti-spoofing (ZSFA)**. Despite the success of face anti-spoofing on known attacks, ZSFA, on the other hand, is a new and unsolved challenge to the community.

The first attempts on ZSFA are Arashloo et al. (2017); Xiong & AbdAlmageed (2018). They address ZSFA between print and replay attacks, and regard it as an outlier detection problem for live faces (a.k.a. real human faces). With handcrafted features, the live faces are modeled via standard generative models, e.g., GMM, auto-encoder. During testing, an unknown attack is detected if it lies outside the estimated live distribution. These ZSFA works have three drawbacks:

Lacking spoof type variety: Prior models are developed w.r.t. print and replay attacks only. The

[1]There is subtle distinction between 1) *unseen attacks*, attack types that are known to algorithm designers so that algorithms could be tailored to them, but their data are unseen during training; 2) *unknown attacks*, attack types that are neither known to designers nor seen during training. We do not differentiate these two cases and term both unknown attacks.

respective feature design may not be applicable to different unknown attacks.

No spoof knowledge: Prior models only use live faces, without leveraging the available known spoof data. While the unknown attacks are different, the known spoof attacks may still provide valuable information to learn the model.

Limitation of feature selection: They use handcrafted features such as LBP to represent live faces, which were shown to be less effective for known spoof detection Li et al. (2016a); Liu et al. (2018c); Patel et al. (2016a); Yang et al. (2014). Recent deep learning models Jourabloo et al. (2018); Liu et al. (2018c) show the advantage of CNN models for face anti-spoofing.

This work aims to address all three drawbacks. Since one ZSFA model may perform differently when the unknown spoof attack is different, it should be evaluated on a wide range of unknown attacks types. In this work, we substantially expand the study of ZSFA from 2 types of spoof attacks to 13 types. Besides print and replay attacks, we include 5 types of 3D mask attacks, 3 types of makeup attacks, and 3 partial attacks. These attacks cover both impersonation spoofing, i.e., attempt to be authenticated as someone else, and obfuscation spoofing, i.e., attempt to cover attacker's own identity. We collect the first face anti-spoofing database that includes these diverse spoof attacks, termed Spoof in the Wild database with Multiple Attack Types (SiW-M), shown in Tab. 3.1.

To tackle the broader ZSFA, we propose a Deep Tree Network (DTN). Assuming there are both homogeneous features among different spoof types and distinct features within each spoof type, a tree-like model is well-suited to handle this case: learning the homogeneous features in the early tree nodes and distinct features in later tree nodes. Without any auxiliary labels of spoof types, DTN learns to partition data in an unsupervised manner. At each tree node, the partition is performed along the direction of the largest data variation. In the end, it clusters the data into several sub-groups at the leaf level, and learns to detect spoof attacks for each sub-group independently, shown in Fig. 3.1. During the testing, a data sample is routed to the most similar leaf node to

produce a binary decision of live vs. spoof.

In summary, our contributions in this work include :

◇ Conduct an extensive study of zero-shot face anti-spoofing on 13 types of spoof attacks;

◇ Propose a Deep Tree Network to learn features hierarchically and detect unknown spoof;

◇ Collect a new database for ZSFA and achieve the state-of-the-art performance on multiple testing protocols.

3.2 Prior Work

Face Anti-spoofing Image-based face anti-spoofing refers to face anti-spoofing techniques that only take RGB images as input without extra information such as depth or heat. In early years, researchers utilize liveness cues, such as eye blinking and head motion, to detect print attacks Kollreider et al. (2007); Pan et al. (2007); Patel et al. (2016a); Shao et al. (2017). However, when encountering unknown attacks, such as photograh with eye portion cut, and video replay, those methods suffer from a total failure. Later, research move to a more general texture analysis and address print and replay attacks. Researchers mainly utilize handcrafted features, e.g., LBP Boulkenafet et al. (2015); de Freitas Pereira et al. (2012, 2013); Määttä et al. (2011), HoG Komulainen et al. (2013a); Yang et al. (2013), SIFT Patel et al. (2016b) and SURF Boulkenafet et al. (2017a), with traditional classifiers, e.g., SVM and LDA, to make a binary decision. Those methods perform well on the testing data from the same database. However, while changing the testing conditions such as lighting and background, they often have a large performance drop, which can be viewed as an overfitting issue. Moreover, they also show limitations in handling 3D mask attacks, mentioned in Liu et al. (2016a).

To overcome the overfitting issue, researchers make various attempts. Boulkenafet et al. extract

the spoofing features in HSV+YCbCR space Boulkenafet et al. (2015). Works in Agarwal et al. (2016); Bao et al. (2009); Bharadwaj et al. (2014); Feng et al. (2016); Xu et al. (2015) consider features in the temporal domain. Recent works Agarwal et al. (2016); Atoum et al. (2017) augment the data by using image patches, and fuse the scores from patches to a single decision. For 3D mask attacks, the heart pulse rate is estimated to differentiate 3D mask from real faces Li et al. (2016b); Liu et al. (2016a). In the deep learning era, researchers propose several CNN works Atoum et al. (2017); Feng et al. (2016); Jourabloo et al. (2018); Li et al. (2016a); Liu et al. (2018c); Patel et al. (2016a); Yang et al. (2014) that outperform the traditional methods.

Zero-shot learning and unknown spoof attacks Zero-shot object recognition, or more generally, zero-shot learning, aims to recognize objects from unknown classes Socher et al. (2013), i.e., object classes unseen in training. The overall idea is to associate the known and unknown classes via a semantic embedding, whose embedding spaces can be attributes Lampert et al. (2009), word vector Frome et al. (2013), text description Zhang et al. (2017a) and human gaze Karessli et al. (2017).

Zero-shot learning for unknown spoof attack, i.e., ZSFA, is a relatively new topic with unique properties. Firstly, unlike zero-shot object recognition, ZSFA emphasizes the detection of spoof attacks, instead of recognizing specific spoof types. Secondly, unlike generic objects with rich semantic embedding, there is no explicit well-defined semantic embedding for spoof patterns Jourabloo et al. (2018). As elaborated in Sec. 3.1, prior ZSFA works Arashloo et al. (2017); Xiong & AbdAlmageed (2018) only model the live data via handcrafted features and standard generative models, with several drawbacks. In this work, we propose a deep tree network to unsupervisely learn the semantic embedding for known spoof attacks. The partition of the data naturally associates certain semantic attributes with the sub-groups. During the testing, the unknown attacks are projected to the embedding to find the closest attributes for spoof detection.

Dataset	Num. of subj./vid.	Face variations			Spoof attack types					Total num. of spoof types
		pose	expression	lighting	replay	print	3D mask	makeup	partial	
CASIA-FASD Zhang et al. (2012)(2012)	50/600	Frontal	No	No	1	2	0	0	0	3
Replay-Attack Chingovska et al. (2012)(2012)	50/1,200	Frontal	No	Yes	1	1	0	0	0	2
HKBU-MARs Liu et al. (2016a)(2016)	35/1,008	Frontal	No	Yes	0	0	2	0	0	2
Oulu-NPU Boulkenafet et al. (2017b)(2017)	55/5,940	Frontal	No	No	1	1	0	0	0	2
SiW Liu et al. (2018c)(2018)	165/4,620	$< 90°$	Yes	Yes	1	1	0	0	0	2
SiW-M(2019)	493/1,630	$< 90°$	Yes	Yes	1	1	5	3	3	13

Table 3.1 Comparing our SiW-M with existing face anti-spoofing datasets.

Deep tree networks Tree structure is often found helpful in tackling language-related tasks such as parsing and translation Chen et al. (2018), due to the intrinsic relation of words and sentences. E.g., tree models are applied to joint vision and language problems such as visual question reasoning Cao et al. (2018). Tree structure also has the property for learning features hierarchically. Face alignment works Kazemi & Sullivan (2014); Valle et al. (2018) utilize the regression trees to estimate facial landmarks from coarse to fine. Xiong et al. propose a tree CNN to handle the large-pose face recognition Xiong et al. (2015). In Kaneko et al. (2018), Kaneko et al. propose a GAN with decision trees to learn hierarchically interpretable representations. In our work, we utilize tree networks to learn the latent semantic embedding for ZSFA.

Face anti-spoofing databases Given the significance of a good-quality database, researchers have released several face anti-spoofing databases, such as CASIA-FASD Zhang et al. (2012), Replay-Attack Chingovska et al. (2012), OULU-NPU Boulkenafet et al. (2017b), and SiW Liu et al. (2018c) for print/replay attacks, and HKBU-MARs Liu et al. (2016a) for 3D mask attacks. Early databases such as CASIA-FASD and Replay-Attack Zhang et al. (2012) have limited subject variety, pose/expression/lighting variations, and video resolutions. Recent databases Boulkenafet et al. (2017b); Liu et al. (2016a, 2018c) improve those aspects, and also set up diverse evaluation protocols. However, up to now, all databases focus on either print/replay attacks, or 3D mask attacks. To provide a comprehensive study of face anti-spoofing, especially the challenging ZSFA, we for the first time collect the database with diverse types of spoof attacks, as in Tab. 3.1. The details of

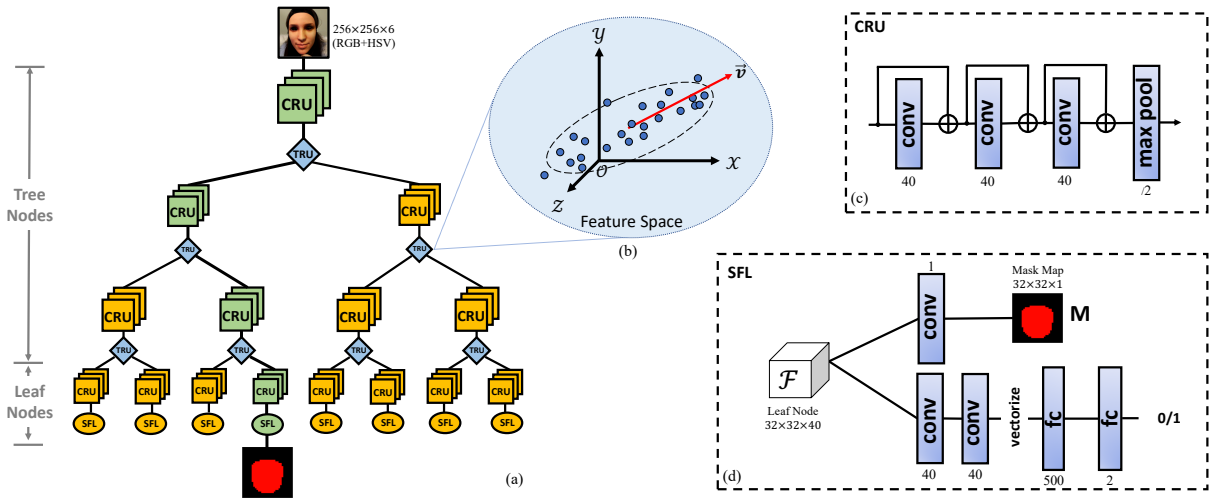


Figure 3.2 The proposed Deep Tree Network (DTN) architecture. (a) the overall structure of DTN. A tree node consists of a Convolutional Residual Unit (CRU) and a Tree Routing Unit (TRU), and a leaf node consists of a CRU and a Supervised Feature Learning (SFL) module. (b) the concept of Tree Routing Unit (TRU): finding the base with largest variations; (c) the structure of each Convolutional Residual Unit (CRU); (d) the structure of the Supervised Feature Learning (SFL) in the leaf nodes.

our database are in Sec. 3.4.

3.3 Deep Tree Network for ZSFA

The main purposes of DTN are twofold: 1) discover the semantic sub-groups for known spoofs; 2) learn the features in a hierarchical way. The architecture of DTN is shown in Fig. 3.2. Each tree node consists of a Convolutional Residual Unit (CRU) and a Tree Routing Unit (TRU), while the leaf node consists of a CRU and a Supervised Feature Learning (SFL) module. CRU is a block with convolutional layers and the short-cut connection. TRU defines a node routing function to route a data sample to one of the child nodes. The routing function partitions all visiting data along the direction with the largest data variation. SFL module concatenates the classification supervision and the pixel-wise supervision to learn the spoofing features.

3.3.1 Unsupervised Tree Learning

3.3.1.1 Node Routing Function

For a TRU node, let's assume the input $\boldsymbol{x} = f(\mathbf{I} \mid \theta) \in \mathbb{R}^m$ is the vectorized feature response, $\mathbf{I}$ is data input, θ is the parameters of the previous CRUs, and $\mathcal{S}$ is the set of data samples $\mathbf{I}_k, k = 1, 2, ..., K$ that visit this TRU node. In Xiong et al. (2015), Xiong et al. define a routing function as:

$$\varphi(\boldsymbol{x}) = \boldsymbol{x}^T \cdot \boldsymbol{v} + \tau, \tag{3.1}$$

where $\boldsymbol{v}$ denotes the projection vector and τ is the bias. Data $\mathcal{S}$ can then be split into $\mathcal{S}_{left}$: $\{\mathbf{I}_k | \varphi(\boldsymbol{x}_k) < 0, \mathbf{I}_k \in \mathcal{S}\}$ and $\mathcal{S}_{right}$: $\{\mathbf{I}_k | \varphi(\boldsymbol{x}_k) \geq 0, \mathbf{I}_k \in \mathcal{S}\}$, and directed to the left and right child node, respectively. To learn this function, they propose to maximize the distance between the mean of $\mathcal{S}_{left}$ and $\mathcal{S}_{right}$, while keeping the mean of $\mathcal{S}$ centered at 0. This unsupervised loss is formulated as:

$$\mathcal{L} = \frac{(\frac{1}{N} \sum\limits_{I_k \in \mathcal{S}} \varphi(\boldsymbol{x}_k))^2}{(\frac{1}{N_l} \sum\limits_{I_k \in \mathcal{S}_{left}} \varphi(\boldsymbol{x}_k) - \frac{1}{N_r} \sum\limits_{I_k \in \mathcal{S}_{right}} \varphi(\boldsymbol{x}_k))^2}, \tag{3.2}$$

where N, N_l, N_r denote the number of samples in each set.

However, in practice, minizing Equ. 3.2 might not lead to a satisfactory solution. Firstly, the loss can be minimized by increasing the norm of either $\boldsymbol{v}$ or $\boldsymbol{x}$, which is a trivial solution. Secondly, even when the norms of $\boldsymbol{v}$, $\boldsymbol{x}$ are constrained, Equ. 3.2 is affected by the density of data $\mathcal{S}$ and can be sensitive to the outliers. In other words, the zero expectation of $\varphi(\boldsymbol{x})$ does not necessarily result in a balanced partition of data $\mathcal{S}$. Local minima could be achieved when all data are spit to one side. In some cases, the tree may suffer from collapsing to a few (even one) leaf nodes.

To better partition the data, we propose a novel routing function and an unsupervised loss.

Regardless of τ, the dot product between $\boldsymbol{x}^T$ and $\boldsymbol{v}$ can be regarded as projecting $\boldsymbol{x}$ to the direction of $\boldsymbol{v}$. We design $\boldsymbol{v}$ such that we can observe the largest variation after projection. Inspired by the concept of PCA, the optimal solution naturally becomes the largest PCA basis of data $\mathcal{S}$. To achieve this, we first constrain $\boldsymbol{v}$ to be norm 1 and reformulate Equ. 3.1 as:

$$\varphi(\boldsymbol{x}) = (\boldsymbol{x} - \boldsymbol{\mu})^T \cdot \boldsymbol{v}, \quad \|\boldsymbol{v}\| = 1, \tag{3.3}$$

where $\boldsymbol{\mu}$ is the mean of data $\mathcal{S}$. Then, finding $\boldsymbol{v}$ is identical to finding the largest eigenvector of the covariance matrix $\bar{\boldsymbol{X}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\mathcal{S}}$, where $\bar{\boldsymbol{X}}_{\mathcal{S}} = \boldsymbol{X}_{\mathcal{S}} - \boldsymbol{\mu}$, and $\boldsymbol{X}_{\mathcal{S}} \in \mathbb{R}^{N \times K}$ is the data matrix. Based on the definition of eigen-analysis $\bar{\boldsymbol{X}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\mathcal{S}} \boldsymbol{v} = \lambda \boldsymbol{v}$, our optimization aims to maximize:

$$\arg\max_{\boldsymbol{v},\theta} \lambda = \arg\max_{\boldsymbol{v},\theta} \boldsymbol{v}^T \bar{\boldsymbol{X}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\mathcal{S}} \boldsymbol{v}. \tag{3.4}$$

The loss for learning the routing function is formulated as:

$$\mathcal{L}_{route} = \exp(-\alpha \boldsymbol{v}^T \bar{\boldsymbol{X}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\mathcal{S}} \boldsymbol{v}) + \beta \mathrm{Tr}(\bar{\boldsymbol{X}}_{\mathcal{S}}^T \bar{\boldsymbol{X}}_{\mathcal{S}}), \tag{3.5}$$

where α, β are scalars, and set as 1e-3, 1e-2 in our experiments. We apply the exponential function on the first term to make the maximization problem bounded. The second term is introduced as a regularizer to prevent trivial solutions by constraining the trace of covariance matrix of $\bar{\boldsymbol{X}}_{\mathcal{S}}$.

3.3.1.2 Tree of Known Spoofs

With the routing function, we can build the entire binary tree. Fig. 3.2 shows a binary tree of depth of 4, with 8 leaf nodes. As mentioned early in Sec. 3.3, the tree is designed to find the semantic sub-groups from all known spoofs, and is termed as spoof tree. Similarly, we may also train live tree

with live faces only, as well as general data tree with both live and spoof data. Compared to spoof tree, live and general data tree have some drawbacks. Live tree does not convey semantic meaning for the spoof, and the attributes learned at each node cannot help to route and better detect spoof; General data tree may result in imbalanced sub-groups, where samples of one class outnumber another. Such imbalance would cause bias for supervised learning in the next stage.

Hence, when we compute Equ. 3.5 to learn the routing functions, we only consider the spoof samples to construct $\boldsymbol{X}_{\mathcal{S}}$. To have a balanced sub-group for each leaf, we suppress the responses of live data to zero, so that all live data can be evenly partitioned to the child nodes. Meanwhile, we also suppress the responses of the spoof data that do not visit this node, so that every node models the distribution of a unique spoof subset.

Formally, for each node, we maximize the routing function responses of spoof data that visit this node (denoted as $\mathcal{S}$), while minimizing the responses of other data (denoted as $\mathcal{S}^{-}$), including all live data and spoof data that don't visit this node, i.e., that visit neighboring nodes. To achieve this objective, we define the following loss:

$$\mathcal{L}_{uniq} = -\frac{1}{N} \sum_{\mathbf{I}_k \in \mathcal{S}} \left\| \bar{\boldsymbol{x}}_k^T \boldsymbol{v} \right\|^2 + \frac{1}{N^-} \sum_{\mathbf{I}_k \in \mathcal{S}^-} \left\| \bar{\boldsymbol{x}}_k^T \boldsymbol{v} \right\|^2 . \tag{3.6}$$

3.3.2 Supervised Feature Learning

Given the routing functions, a data sample $\mathbf{I}_k$ will be assigned to one of the leaf nodes. Let's first define the feature output of leaf node as $\mathcal{F}(\mathbf{I}_k \,|\, \theta)$, shortened as $\mathcal{F}_k$ for simplicity. At each leaf node, we define two node-wise supervised tasks to learn discriminative features: 1) binary classification drives the learning of a high-level understanding of live vs. spoof faces, 2) pixel-wise mask regression draws CNN's attention to low-level local feature learning.

Classification supervision To learn a binary classifier, as shown in Fig. 3.2(d), we apply two additional convolution layers and two fully connected layers on $\mathcal{F}_k$ to generate a feature vector $\mathbf{c}_k \in \mathbb{R}^{500}$. We supervise the learning via the softmax cross entropy loss:

$$\mathcal{L}_{class} = \frac{1}{N} \sum_{I_k \in \mathcal{S}} \Big\{ (1 - y_k) \log(1 - p_k) - y_k \log p_k \Big\} \tag{3.7}$$

$$p_k = \frac{\exp(\mathbf{w}_1{}^T \mathbf{c}_k)}{\exp(\mathbf{w}_0{}^T \mathbf{c}_k) + \exp(\mathbf{w}_1{}^T \mathbf{c}_k)}, \tag{3.8}$$

where $\mathcal{S}$ represents all the data samples that arrive this leaf node, N denotes the number of samples in $\mathcal{S}$, $\{\mathbf{w}_0, \mathbf{w}_1\}$ are the parameters in the last fully connected layer, and y_k is the label of data sample k (1 denotes spoof, and 0 live).

Pixel-wise supervision We also concatenate another convolution layer to $\mathcal{F}_k$ to generate a map response $\mathbf{M}_k \in \mathbb{R}^{32 \times 32}$. Inspired by the prior work Liu et al. (2018c), we leverage the semantic prior knowledge of face shapes and spoof attack position to provide a pixel-wise supervision. Using the dense face alignment model Liu et al. (2017), we provide a binary mask $\mathbf{D}_k \in \mathbb{R}^{32 \times 32}$, shown in Fig. 3.3, to indicate the pixels of spoof mediums. Thus, for a leaf node, the loss function for the pixel-wise supervision is:

$$\mathcal{L}_{mask} = \frac{1}{N} \sum_{I_k \in \mathcal{S}} \left\| \mathbf{M}_k - \mathbf{D}_k \right\|_1 . \tag{3.9}$$

Overall loss Finally, we apply the supervised losses on p leaf nodes, the unsupervised losses on q TRU nodes, and formulate our training loss as:

$$\mathcal{L} = \sum_{i=1}^{p} (\alpha_1 \mathcal{L}^i_{class} + \alpha_2 \mathcal{L}^i_{mask}) + \sum_{j=1}^{q} (\alpha_3 \mathcal{L}^j_{route} + \alpha_4 \mathcal{L}^j_{uniq}), \tag{3.10}$$

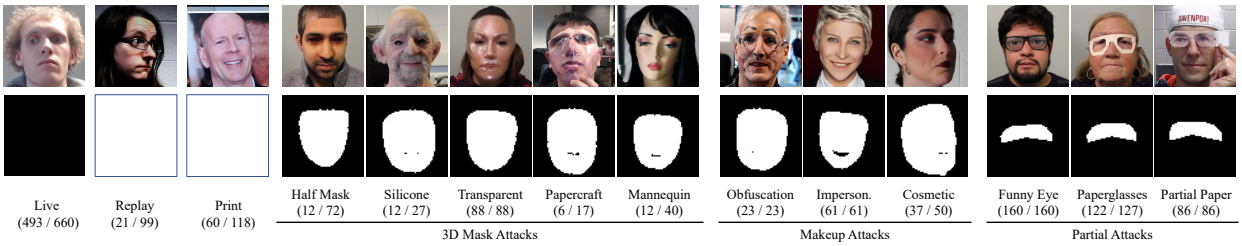


Figure 3.3 The examples of the live faces and 13 types of spoof attacks. The second row shows the ground truth masks for the pixel-wise supervision $\mathbf{D}_k$. For (m, n) in the third row, m/n denotes the number of subjects/videos for each type of data.

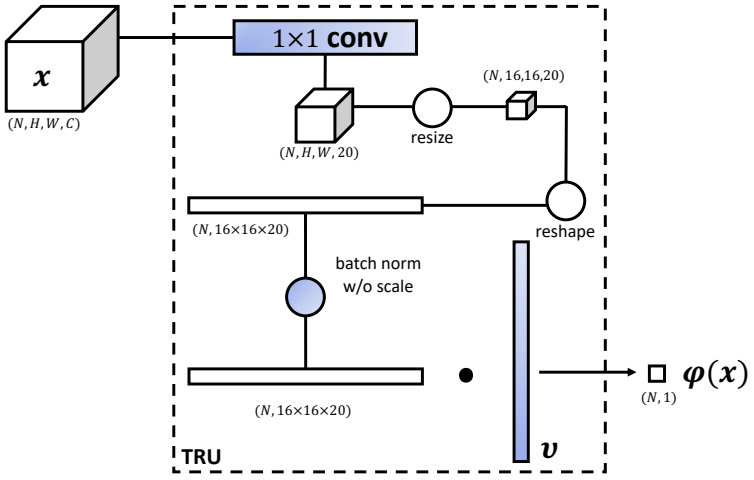


Figure 3.4 The structure of the Tree Routing Unit (TRU).

where α_1,α_2,α_3,α_4 are the regularization coefficients for each term, and are set as 0.001, 1.0, 2.0, 0.001 respectively. For a 4-layer DTN, $p = 8$ and $q = 7$.

3.3.3 Network Architecture

Deep Tree Network (DTN) DTN is the main framework of the proposed model. It takes $\mathbf{I} \in \mathbb{R}^{256\times256\times6}$ as input, where the 6 channels are RGB+HSV color spaces. We concatenate three 3×3 convolution layers with 40 channels and 1 max-pooling layer, and group them as one Convolutional Residual Unit (CRU). Each convolution layer is equipped with ReLU and group normalization layer Wu & He (2018), due to the dynamic batch size in the network. We also apply a shortcut connection for each convolution layer. For each tree node, we deploy one CRU before the TRU. At

the leaf node, DTN produces the feature representation of input $\mathbf{I}$ as $\mathcal{F}(\mathbf{I} \mid \theta) \in \mathbb{R}^{32\times32\times40}$, then uses one 1×1 convolution layer to generate the binary mask map $\mathbf{M}$.

Tree Routing Unit (TRU) TRU is the module routing the data sample to one of the child CRUs. As shown in Fig. 3.4, it first compresses the feature by using an 1×1 convolution layer, and resizing the response spatially. For the root node, we compress the CRU feature to $\mathbf{x} \in \mathbb{R}^{32\times32\times10}$, and for later tree node, we compress the CRU feature to $\mathbf{x} \in \mathbb{R}^{16\times16\times20}$. Compressing the input feature to a smaller size helps to reduce the burden of computating and saving the covariance matrix in Equ. 3.5. E.g., the vectorized feature for the first CRU is $\boldsymbol{x} \in \mathbb{R}^{655,360}$, and the covariance matrix of $\boldsymbol{x}$ can take ~ 400GB in memory. However, after compression the vectorized feature is $\boldsymbol{x} \in \mathbb{R}^{10,240}$, and the covariance matrix of $\boldsymbol{x}$ only needs ~ 0.1GB of memory.

After that, we vectorize the output and apply the routing function $\varphi(\boldsymbol{x})$. To compute $\boldsymbol{\mu}$ in Equ. 3.3, instead of optimizing it as a variable of the network, we simply apply a batch normalization layer without scaling to save the moving average of each mini-batch. In the end, we project the compressed CRU response to the largest basis $\boldsymbol{v}$ and obtain the projection coefficient. Then we assign the samples with negative coefficient to the left child CRU and the samples with positive coefficient to the right child CRU.

Implementation details With the overall loss in Equ. 3.10, our proposed network is trained in an end-to-end fashion. All losses are computed based on each mini-batch. DTN modules and TRU modules are optimized alternately. While optimizing DTN, we keep the parameters of TRUs fixed and vice versa.

3.4 Spoof in the Wild Database with Multiple Attack Types

To benchmark face anti-spoofing methods specifically for unknown attacks, we collect the Spoof in the Wild database with Multiple Attack Types (SiW-M). Compared with the previous databases in Tab. 3.1, SiW-M shows a great diversity in spoof attacks, subject identities, environments and other factors.

For spoof data collection, we consider two spoofing scenarios: *impersonation*, which entails the use of spoof to be recognized as someone else, and *obfuscation*, which entails the use to remove the attacker's own identity. In total, we collect 968 videos of 13 types of spoof attacks listed hieratically in Fig 3.3. For all 5 mask attacks, 3 partial attacks, obfuscation makeup and cosmetic makeup, we record 1080P HD videos. For impersonation makeup, we collect 720P videos from Youtube due to the lack of special makeup artists. For print and replay attacks, we intend to collect videos from harder cases where the existing system fails. Hence, we deploy an off-the-shelf face anti-spoofing algorithm Liu et al. (2018c) and record spoof videos when the algorithm predicts live.

For live data, we include 660 videos from 493 subjects. In comparison, the number of subjects in SiW-M is 9 times larger than Oulu-NPU Boulkenafet et al. (2017b) and CASIA-FASD Zhang et al. (2012), and 3 times larger than SiW Liu et al. (2018c). In addition, subjects are diverse in ethnicity and age. The live videos are collected in 3 sessions: $1)$ a room environment where the subjects are recorded with few variations such as pose, lighting and expression (PIE). $2)$ a different and much larger room where the subjects are also recorded with PIE variations. $3)$ a mobile phone mode, where the subjects are moving while the phone camera is recording. Extreme pose angles and lighting conditions are introduced. Similar to print and replay videos, we deploy the face anti-spoofing algorithm Liu et al. (2018c) to find out the videos where the algorithm predicts spoof. Hence, this third session is a harder scenario.

In total, we collect 1,630 videos and each lasts 5-7 seconds. The 1080P videos are recorded by Logitech C920 webcam and Canon EOS T6. To use SiW-M for the study of ZSFA, we define the leave-one-out testing protocols. Each time we train a model with 12 types of spoof attacks plus the 80% of the live videos, and test on the left 1 attack type plus the 20% of live videos. There is no overlapping subjects between the training and testing sets of live videos.

3.5 Experimental Results

3.5.1 Experimental Setup

Databases We evaluate our proposed method on multiple databases. We deploy the leave-one-out testing protocols on SiW-M and report the results of 13 experiments. Also, we test on previous face anti-spoofing databases, including CASIA Zhang et al. (2012), Replay-Attack Chingovska et al. (2012), and MSU-MFSD Wen et al. (2015)), compare with the state of the art.

Evaluation metrics We evaluate with the following metrics: Attack Presentation Classification Error Rate (APCER) ISO/IEC-JTC-1/SC-37 (2016), Bona Fide Presentation Classification Error Rate (BPCER) ISO/IEC-JTC-1/SC-37 (2016), the average of APCER and BPCER, Average Classification Error Rate (ACER) ISO/IEC-JTC-1/SC-37 (2016), Equal Error Rate (EER), and Area Under Curve (AUC). Note that, in the evaluation of unknown attacks, we assume there is no validation set to tune the model and thresholds while calculating the metrics. Hence, we determine the threshold based on the training set and fix it for all testing protocols. A single test sample is one video frame, instead of one video.

Parameter setting The proposed method is implemented in Tensorflow Abadi et al. (2016), and trained with a constant learning rate of 0.001 with a batch size of 32. It takes 15 epochs to converge. We randomly initialize all the weights using a normal distribution of 0 mean and 0.02 standard

deviation.

3.5.2 Experimental Comparison

3.5.2.1 Ablation Study

All ablation studies use the Funny Eye protocol.

Different fusion methods In the proposed model, both the norm of the mask maps and binary spoof scores could be utilized for the final classification. To find the best fusion method, we compute ACER from using map norm, softmax score, the maximum of map norm and softmax score, and the average of two values, and obtain 31.7%, 20.5%, 21.0%, and 19.3% respectively. Since the average score of the mask norm and binary spoof score performs the best, we use it for the remaining experiments. Moreover, we set 0.2 as the final threshold to compute APCER, BPCER and ACER for all the experiments.

Different routing methods Routing is a crucial step to find the best subgroup to detect spoofness of a testing sample. To show the effect of proper routing, we evaluate 2 alternative routing strategies: random routing and pick-one-leaf. Random routing denotes randomly selecting one leaf node for a testing sample to produce prediction; Pick-one-leaf denotes constantly selecting one particular leaf node to produce results, for which we report the mean score and standard deviation of 8 selections. Shown in Tab. 3.2, both strategies perform worse than the proposed routing function. In addition, the large standard deviation of pick-one-leaf strategy shows the *large* performance difference of 8 subgroups on the *same type* of unknown attacks, and demonstrates the necessity of a proper routing.

Advantage of each loss function We have three important designs in our unsupervised tree learning: route loss $\mathcal{L}_{route}$, data used to compute the route loss, and the unique loss $\mathcal{L}_{uniq}$. To

Strategies	APCER	BPCER	ACER	EER
Random routing	37.1	**16.1**	26.6	24.7
Pick-one-leaf	51.2 ± 20.0	18.1 ± 4.9	34.7 ± 8.8	24.1 ± 3.1
Proposed routing function	**17.0**	21.5	**19.3**	**19.8**

Table 3.2 Compare models with different routing strategies.

Methods	APCER	BPCER	ACER	EER
MPT Xiong et al. (2015)	31.4	24.2	27.8	27.3
Live data $\surd$, Spoof data $\surd$, Unique Loss $\times$	**1.4**	73.3	37.3	31.2
Live data $\times$, Spoof data $\surd$, Unique Loss $\times$	70.0	12.7	41.3	44.8
Live data $\surd$, Spoof data $\surd$, Unique Loss $\surd$	54.2	**12.5**	33.4	36.2
Live data $\times$, Spoof data $\surd$, Unique Loss $\surd$	17.0	21.5	**19.3**	**19.8**

Table 3.3 Compare models with different tree losses and strategies. The first two terms of row 2-5 refer to using live or spoof data in tree learning. The last row is our method.

Methods	CASIA Zhang et al. (2012)			Replay-Attack Chingovska et al. (2012)			MSU Wen et al. (2015)			Overall
	Video	Cut Photo	Warped Photo	Video	Digital Photo	Printed Photo	Printed Photo	HR Video	Mobile Video	
OC-SVM$_{RBF}$+BSIF Arashloo et al. (2017)	70.7	60.7	95.9	84.3	88.1	73.7	64.8	87.4	74.7	78.7 ± 11.7
SVM$_{RBF}$+LBP Boulkenafet et al. (2017b)	91.5	91.7	84.5	99.1	98.2	87.3	47.7	99.5	**97.6**	88.6 ± 16.3
NN+LBP Xiong & AbdAlmageed (2018)	**94.2**	88.4	79.9	99.8	95.2	78.9	50.6	99.9	93.5	86.7 ± 15.6
Ours	90.0	**97.3**	**97.5**	**99.9**	**99.9**	**99.6**	**81.6**	**99.9**	97.5	$\mathbf{95.9 \pm 6.2}$

Table 3.4 AUC (%) of the model testing on CASIA, Replay, and MSU-MFSD.

show the effect of each loss and the training strategy, we train and compare networks with each loss excluded and alternative strategies. First, we train a network with the routing function proposed in Xiong et al. (2015), and then 4 models with different modules on and off, shown in Tab. 3.3. The model with MPT Xiong et al. (2015) routes data only to 2 leaf nodes out of 8 (i.e. tree collapse issue), which limits the performance. Models without the unique loss exhibit the imbalance routing issue where sub-groups cannot be trained properly. Models using all data to learn the tree show worse performances than using spoof data only. Finally, the proposed method performs the best among all options.

3.5.2.2 Testing on existing databases

Following the protocol proposed in Arashloo et al. (2017), we use CASIA Zhang et al. (2012), Replay-Attack Chingovska et al. (2012) and MSU-MFSD Wen et al. (2015) to perform ZSFA

testing between replay and print attacks. Tab. 3.4 compares the proposed method with top three methods selected from over 20 methods in Arashloo et al. (2017); Boulkenafet et al. (2017b); Xiong & AbdAlmageed (2018). Our proposed method outperforms the prior state of the art by a convincing margin of 7.3%, and our smaller standard deviation further indicates a consistently good performance among unknown attacks.

3.5.2.3 Testing on SiW-M

We execute 13 leave-one-out testing protocols on SiW-M. We compare with two of the most recent face anti-spoofing methods Boulkenafet et al. (2017b); Liu et al. (2018c), and set Liu et al. (2018c) as the baseline, which has demonstrated its SOTA performance on various benchmarks. For a fair comparison with the baseline, we provide the same pixel-wise labeling (as in Fig. 3.3), and set the same threshold of 0.2 to compute APCER, BPCER, and ACER.

As shown in Tab. 3.5, our method achieves an overall better APCER, ACER and EER, with the improvement of baseline by 55%, 29%, and 5%. Specifically, we reduce the ACERs of transparent mask, funny eye, and paper glasses by 31%, 61%, and 51%, where the baseline models can be considered as total failures since they recognize most of the attacks as live. Note that, ACER is more valuable in the context of ZSFA: no evaluation data for setting threshold and considerably varied thresholds for obtaining the EER performance. For instance, EERs of paper glasses model are similar between the baseline and our method, but with a preset threshold, our method offers a much better ACER.

Moreover, the proposed method is a more compact model thanLiu et al. (2018c). Given the input size of $256 \times 256 \times 6$, the baseline requires 87 GFlops to compute the result while our method only needs 6 GFlops ($\times 15$ smaller). More analysis are shown with visualization in Sec. 3.5.2.4.

Among all the attacks, replay, print, half mask, paper mask, impersonation makeup are im-

Methods	Metrics(%)	Replay	Print	Mask Attacks					Makeup Attacks			Partial Attacks			Average
				Half	Silicone	Trans.	Paper	Manne.	Obfusc.	Imperson.	Cosmetic	Funny Eye	Paper Glasses	Paper	
SVM$_{RBF}$+LBP Boulkenafet et al. (2017b)	APCER	19.1	15.4	40.8	20.3	70.3	**0.0**	4.6	96.9	35.3	**11.3**	53.3	58.5	0.6	32.8 ± 29.8
	BPCER	22.1	21.5	21.9	21.4	20.7	23.1	22.9	21.7	12.5	22.2	18.4	20.0	22.9	21.0 ± 2.9
	ACER	20.6	18.4	31.3	21.4	45.5	11.6	13.8	59.3	23.9	16.7	35.9	39.2	11.7	26.9 ± 14.5
	EER	20.8	18.6	36.3	21.4	37.2	7.5	14.1	51.2	19.8	16.1	34.4	33.0	7.9	24.5 ± 12.9
AuxiliaryLiu et al. (2018c)	APCER	23.7	7.3	27.7	**18.2**	97.8	8.3	16.2	100.0	18.0	16.3	91.8	72.2	0.4	38.3 ± 37.4
	BPCER	**10.1**	**6.5**	**10.9**	**11.6**	**6.2**	**7.8**	**9.3**	11.6	**9.3**	**7.1**	**6.2**	**8.8**	**10.3**	$\mathbf{8.9 \pm 2.0}$
	ACER	16.8	6.9	19.3	**14.9**	52.1	8.0	12.8	55.8	13.7	**11.7**	49.0	40.5	**5.3**	23.6 ± 18.5
	EER	14.0	4.3	**11.6**	**12.4**	**24.6**	7.8	10.0	72.3	10.1	**9.4**	21.4	**18.6**	**4.0**	17.0 ± 17.7
Ours	APCER	**1.0**	**0.0**	**0.7**	24.5	**58.6**	0.5	**3.8**	**73.2**	**13.2**	12.4	**17.0**	**17.0**	**0.2**	$\mathbf{17.1 \pm 23.3}$
	BPCER	18.6	11.9	29.3	12.8	13.4	8.5	23.0	**11.5**	9.6	16.0	21.5	22.6	16.8	16.6 ± 6.2
	ACER	**9.8**	**6.0**	**15.0**	18.7	**36.0**	**4.5**	**7.7**	**48.1**	**11.4**	14.2	**19.3**	**19.8**	8.5	$\mathbf{16.8 \pm 11.1}$
	EER	**10.0**	**2.1**	14.4	18.6	26.5	**5.7**	**9.6**	**50.2**	**10.1**	13.2	**19.8**	20.5	8.8	$\mathbf{16.1 \pm 12.2}$

Table 3.5 The evaluation and comparison of the testing on SiW-M.

personation attacks. The average ACER/EER of impersonation attacks is 9.3/8.5, which is lower than the overall average ACER/EER. This shows that the proposed method handles impersonation attacks better. When the attackers try to impersonate someone, the spoof face is required to be similar as a live face, thus the network can extract anti-spoofing feature more easily. However, when the attackers just try to hidden its own identity (obfascation attacks), the spoof face is not necessary to be look a live face, which is easier to become an outlier of the data distribution and false the system.

3.5.2.4 Visualization and Analysis

To provide a better understanding of the tree learning and ZSFA, we visualize the results in several ways. First, we illustrate the tree routing results. In Fig. 3.5, we rank the spoof data based on the routing function values $\varphi(\boldsymbol{x})$, and provide 8 examples with responses from the smallest to the largest. This offers us an intuitive understanding of what are learned at each tree node. We observe an obvious spoof style transfer: for the first two-layer nodes N_1, N_2 and N_3, the transfer captures the change of general spoof attributes such as image quality and color temperature; for the third-layer tree nodes N_4, N_5, N_6, and N_7, the transfer involves more spoof type specific changes. E.g., N_7 transfers from eye portion spoofs to full face 3D mask spoofs.

Further, Fig. 3.6 quantitatively analyzes the tree routing distributions of all types of data. We

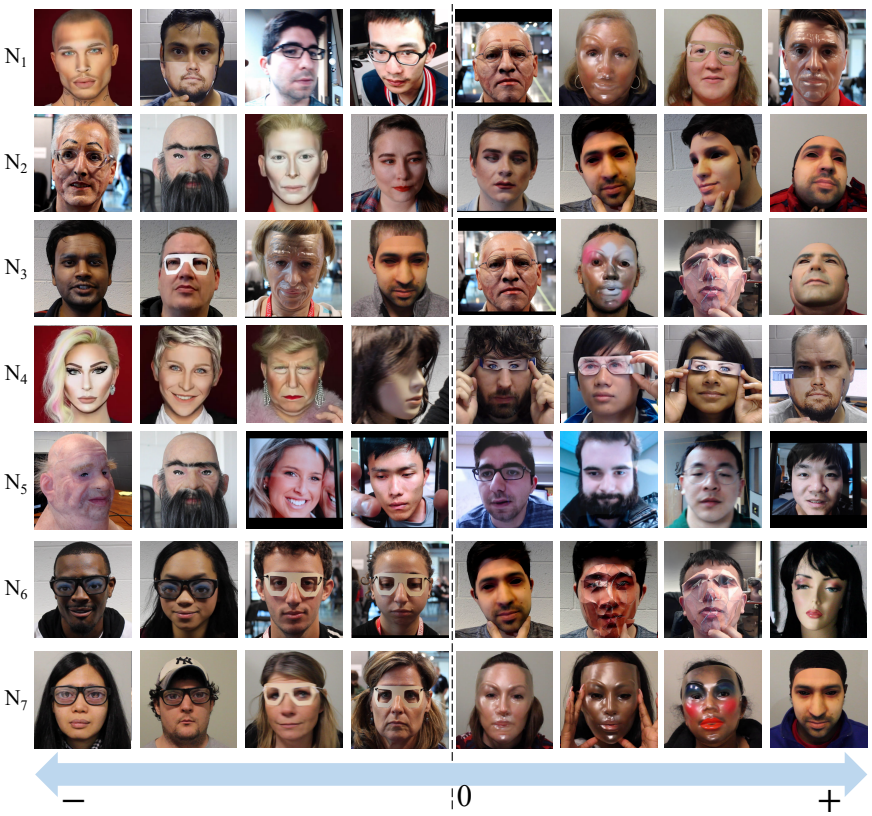


Figure 3.5 Visulization of the Tree Routing.

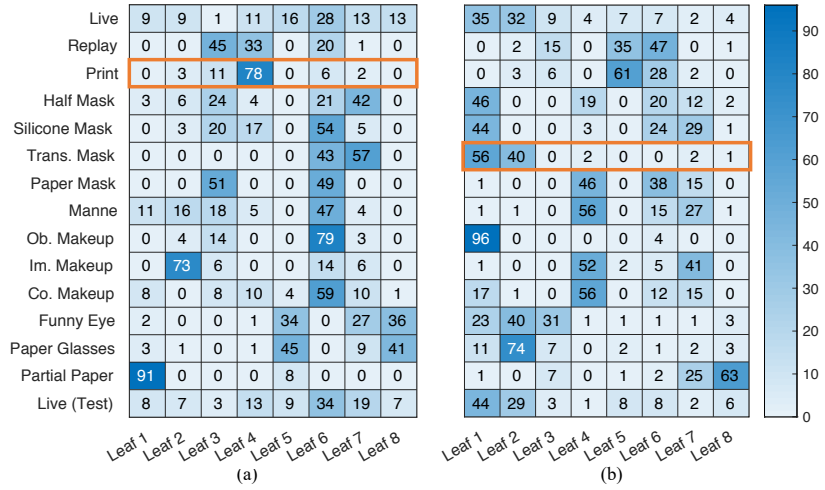


Figure 3.6 Tree routing distribution of live/spoof data. X-axis denotes 8 leaf nodes, and y-axis denotes 15 types of data. The number in each cell represents the percentage (%) of data that fall in that leaf node. Each row is sum to 1. (a) Print Protocol. (b) Transparent Mask Protocol. Yellow box denotes the unknown attacks.

utilize two models, Print and Trans. Mask, to generate the distributions. It can be observed that live samples are relatively more spread out to 8 leaf nodes while the spoof attacks are routed to fewer

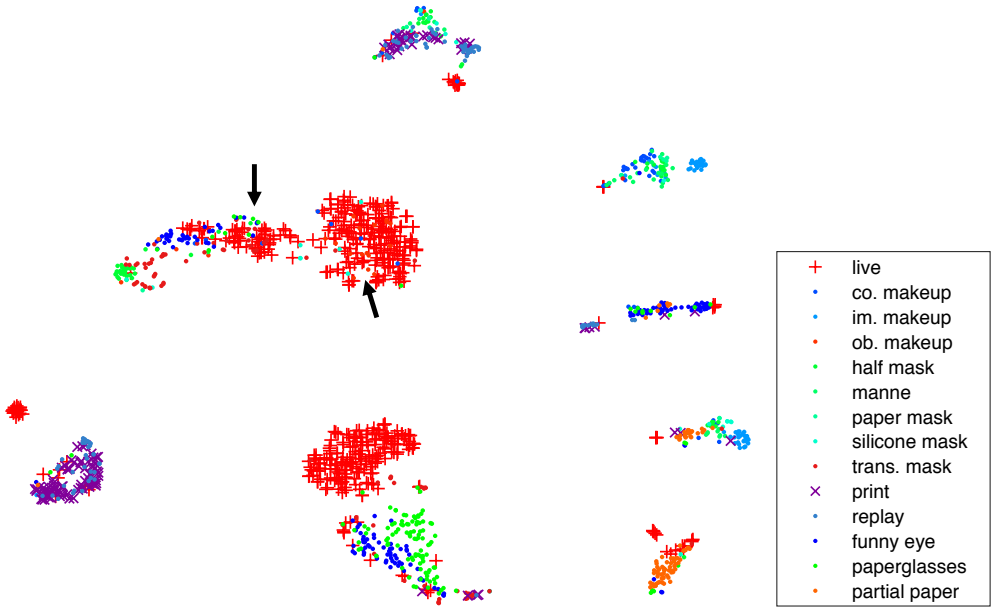


Figure 3.7 t-SNE Visualization of the DTN leaf features.

specific leaf nodes. Two distributions in Fig. 3.6 (a)&(b) share similar semantic sub-groups, which demonstrates the success of the proposed method on learning a tree. E.g., in both models, about half of trans. mask samples share the same leaf node as ob. makeup. By comparing two distributions, most testing unknown spoofs in both models are successfully routed to the most similar sub-groups.

In addition, we use t-SNE Maaten & Hinton (2008) to visualize the feature space of Print model. The t-SNE is able to project the output of the leaf node $\mathcal{F}(\mathbf{I} \mid \theta) \in \mathbb{R}^{32\times32\times40}$ to 2D by preserving the KL divergence distance. Fig. 3.7 shows the features of different types of spoof attacks are well-clustered into 8 semantic sub-groups even though we don't provide any auxiliary labels. Based on these sub-groups, the features of unknown print attacks are well lied in the sub-group of replay and silicone mask, and thus are recognized as spoof. Moreover, with the visualization, we can explain the performance variation among different spoof attacks, shown in Tab. 3.5. Among all, the performance of trans. mask, funny eye, paper glasses and ob. makeup are worse than other protocols. The feature space shows that the live samples lies much closer to those attacks than others ("$\rightarrow$" places), and hence it's harder to distinguish them with the live samples. This demonstrates the

diverse property of different unknown attacks and the necessity of such a wide range evaluation.

3.6 Conclusions

This chapter tackles the zero-shot face antispoofing problem among 13 types of spoof attacks. The proposed method leverages a deep tree network to route the unknown attacks to the most proper leaf node for spoof detection. The tree is trained in an unsupervised fashion to find the feature base with the largest variation to split the spoof data. We collect SiW-M that contains more subjects and spoof types than any previous databases. Finally, we experimentally show superior performance of the proposed method.

Chapter 4

Visualization: Disentangling Spoof Traces with Physical Modeling

4.1 Introduction

As most face recognition systems are based on a monocular RGB camera, monocular RGB based face anti-spoofing has been studied for over a decade, and one of the most common approaches is based on texture analysis in Boulkenafet et al. (2015, 2016); Patel et al. (2016b). Researchers noticed that presenting faces from spoof mediums introduces special texture differences, such as color distortions, unnatural specular highlights, Moir patterns, *etc*. Those texture differences are inherent within spoof mediums and thus hard to remove or camouflage. Conventional approaches build a feature extractor plus classifier pipeline, such as LBP+SVM and HOG+SVM in de Freitas Pereira et al. (2012); Komulainen et al. (2013a), and show good performance on several small databases with constraint environments. In recent years, many works such as Atoum et al. (2017); Liu et al. (2018c, 2019a); Shao et al. (2019a); Yang et al. (2019a) leverage deep learning techniques and show great progress in face anti-spoofing performance. Deep learning based methods can be generally grouped into 3 categories: direct FAS, auxiliary FAS, and generative FAS, as illustrated in Fig. 4.2. Early works Xu et al. (2015); Yang et al. (2014) build vanilla CNN with binary output to directly predict the spoofness of an input face (Fig.4.2a). Methods Liu et al. (2018c); Yang et al. (2019a) propose to learn an intermediate representation, *e.g.*, depth, rPPG, reflection, instead of binary

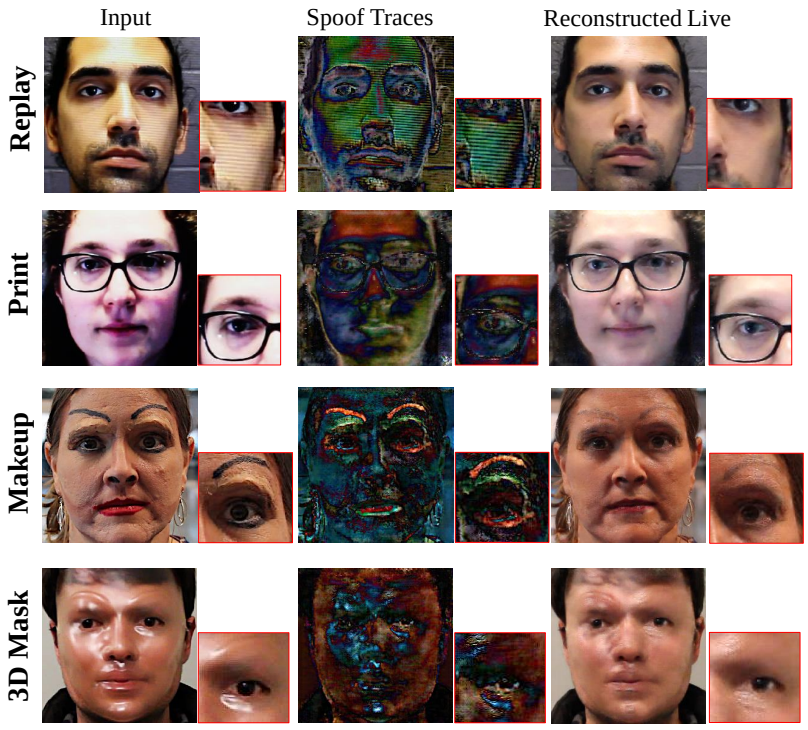
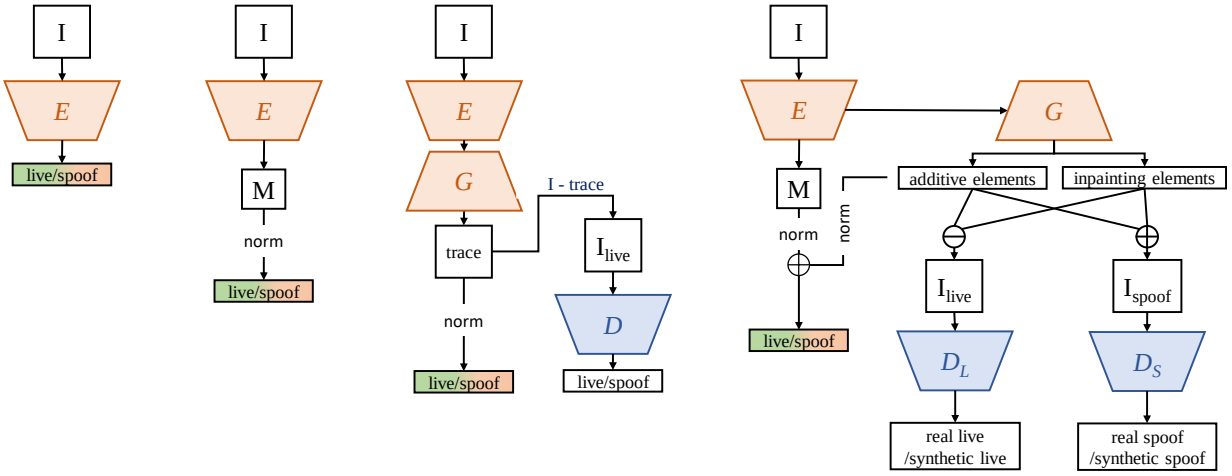


Figure 4.1 The proposed approach can detect spoof faces, disentangle the spoof traces, and reconstruct the live counterparts. It can be applied to diverse spoof types and estimate distinct traces (*e.g.*, Moir pattern in replay attack, artificial eyebrow and wax in makeup attack, color distortion in print attack, and specular highlights in 3D mask attack). Zoom in for details.

classes, which can lead to better generalization and performance (Fig.4.2b). Feng et al. (2020); Jourabloo et al. (2018); Stehouwer et al. (2020) additionally attempt to generate the visual patterns existing in the spoof samples (Fig.4.2c), providing a more intuitive interpretation of the sample's spoofness.

Despite the success, there are still at least three unsolved problems in the topic of deep learning-based face anti-spoofing. First, most prior works are designed to tackle *limited spoof types*, either print/replay or 3D mask solely, while a real-world anti-spoofing system may encounter a wide variety of spoof types including print, replay, various 3D masks, facial makeup, and even unseen attack types. Therefore, to better reflect real-world performance, we need a benchmark to evaluate face anti-spoofing under known attacks, unknown attacks, and their combination (termed *open-set* setting). Second, many approaches formulate face anti-spoofing as a classification/regression



(a) direct FAS (b) auxiliary FAS (c) generative FAS (d) proposed method

Figure 4.2 The comparison of different deep-learning based face anti-spoofing. (a) direct FAS only provides a binary decision of spoofness; (b) auxiliary FAS can provide simple interpretation of spoofness. M denotes the auxiliary task, such as depth map estimation; (c) generative FAS can provide more intuitive interpretation of spoofness, but only for a limited number of spoof attacks; (d) the proposed method can provide spoof trace estimation for generic face spoof attacks.

problem, with a single score as the output. Although auxiliary FAS and generative FAS attempt to offer some extent of interpretation by fixation, saliency, or noise analysis, there is little understanding on what the exact differences are between live and spoof, and what patterns the classifier's decision is based upon. A better interpretation can be estimating the exact patterns differentiating a spoof face and its live counterpart, termed **spoof trace**. Thirdly, compared with other face analysis tasks such as recognition or alignment, the data for face anti-spoofing has several limitations. Most FAS databases are captured in the constraint indoor environment, which has limited intra-subject variation and environment variation. For some special spoof types such as makeup and customized silicone mask, they require highly skilled experts to apply or create, with high cost, which results in very limited samples (*i.e.*, long-tail data). Thus, how to learn from data with limited variations or samples is a challenge for FAS.

In this work, we aim to design a face anti-spoofing model that is applicable to a wide variety of spoof types, termed **generic face anti-spoofing**. We equip this model with the ability to explicitly

disentangle the spoof traces from the input faces. Some examples of spoof trace disentanglement are shown in Fig. 4.1. This is a challenging objective due to the diversity of spoof traces and the lack of ground truth during model learning. However, we believe that fulfilling this objective can bring several benefits:

1. Binary classification for face anti-spoofing would harvest any cue that helps classification, which might include spoof-irrelevant cues such as lighting, and thus hinder generalization. In contrast, spoof trace disentanglement explicitly tackles the most fundamental cue in spoofing, upon which the classification can be more grounded and witness better generalization.

2. With the trend of pursuing explainable AI as mentioned in Arrieta et al. (2020); Turek (2016), it is desirable for the face anti-spoofing model to generate the spoof patterns that support its binary decision, since spoof trace serves as a good visual explanation of the model's decision. Certain properties (*e.g.*, severity, methodology) of spoof attacks might potentially be revealed from the traces.

3. Disentangled spoof traces can enable the synthesis of realistic spoof samples, which addresses the issue of limited training data for the minority spoof types, such as special 3D masks and makeup.

As shown in Fig. 4.2d, we propose a Physics-guided Spoof Trace Disentanglement (PhySTD) to explore the spoof traces for generic face anti-spoofing. To model all types of spoofs, we formulate the spoof trace disentanglement as a combination of *additive* process and *inpainting* process. Additive process describes spoofing as spoof material introducing extra patterns (*e.g.*, moire pattern), where the live counterpart can be recovered by removing those patterns. Inpainting process describes spoofing as spoof material fully covering certain regions of the original face, where the live counterpart of those regions has to be "guessed" as shown in Bertalmio et al. (2000); Liu &

Shu (2015). We further decompose the spoof traces into frequency-dependent components, so that traces with different frequency properties can be equally handled. For the network architecture, we extend a backbone network for auxiliary FAS with a decoder to perform the disentanglement. With no ground truth of spoof traces, we adopt an overall GAN-based training strategy. The generator takes an input face, estimates its spoofness, and disentangles the spoof trace. After obtaining the spoof trace, we can reconstruct the live counterpart from the spoof and synthesize new spoof from the live. The synthesized samples are then sent to multiple discriminators with real samples for adversarial training. The synthesized spoof samples are further utilized to train the generator in a fully supervised fashion, thanks to disentangled spoof traces as ground truth for the synthesized samples. To correct possible geometric discrepancy during spoof synthesis, we propose a novel 3D warping layer to deform spoof traces toward the target live face.

A preliminary version of this work was published in the Proceedings European Conference on Computer Vision (ECCV) 2020 Liu et al. (2020). We extend the work from three aspects. 1) Guided by the physics of how a spoof is generated, we introduce a spoof generation function (SGF) to model the spoof trace disentanglement as a combination of additive and inpainting processes. SGF has a better and more natural modeling of generic spoof attacks, such as paper glass. 2) Previous trace components $\{\mathbf{S}, \mathbf{B}, \mathbf{C}, \mathbf{T}\}$ are not supervised hierarchically so that there exists semantic ambiguity. In this work, we introduce several hierarchical designs in the GAN framework to remedy such ambiguity. 3) We propose an open-set testing scenario to further evaluate the real-world performance for face anti-spoofing models. Both known and unknown attacks are included in the open-set testing. We perform a side-by-side comparison between the proposed approach and the state-of-the-art (SOTA) face anti-spoofing solutions on multiple datasets and protocols.

In summary, the main contributions of this work are as follows:

- We for the first time study spoof trace for *generic* face anti-spoofing, where a wide variety of

spoof types are tackled with one unified framework;

- We propose a novel physics-guided model to disentangle spoof traces, and utilize the spoof traces to synthesize new data samples for enhanced training;
- We propose novel protocols for a generic open-set face anti-spoofing;
- We achieve SOTA anti-spoofing performance and provide convincing visualization for a wide variety of spoof types.

4.2 Related Work

Face Anti-Spoofing Face anti-spoofing has been studied for more than a decade and its development can be roughly divided into three stages. In the early years, researchers leverage spontaneous human movement, such as eye blinking and head motion, to detect simple print photograph or static replay attacks Kollreider et al. (2007); Pan et al. (2007). However, when facing counter attacks, such as print face with eye region cut, and replaying a face video, those methods would fail. In the second stage, researchers pay more attention to texture differences between live and spoof, which are inherent to spoof mediums. Researchers mainly extract hand-crafted features from the faces, *e.g.*, LBP Boulkenafet et al. (2015); de Freitas Pereira et al. (2012, 2013); Määttä et al. (2011), HoG Komulainen et al. (2013a); Yang et al. (2013), SIFT Patel et al. (2016b) and SURF Boulkenafet et al. (2016), and train a classifier to split the live *vs.* spoof, *e.g.*, SVM and LDA.

Recently, face anti-spoofing solutions equipped with deep learning techniques have demonstrated significant improvements over the conventional methods. Methods in Feng et al. (2016); Li et al. (2016a); Patel et al. (2016a); Yang et al. (2014) train a deep neural network to learn a binary classification between live and spoof. In Atoum et al. (2017); Liu et al. (2018c, 2019a); Shao et al.

(2019a); Yang et al. (2019a), additional supervisions, such as face depth map and rPPG signal, are utilized to help the network to learn more generalizable features. As the latest approaches achieving saturated performance on several benchmarks, researchers start to explore more challenging cases, such as few-shot/zero-shot face anti-spoofing Liu et al. (2019a); Qin et al. (2019); Zhao et al. (2019) and domain adaptation in face anti-spoofing Shao et al. (2019a,b).

In this work, we aim to solve an interesting yet very challenging problem: disentangling and visualizing the spoof traces from an input face. A related work Jourabloo et al. (2018) also adopts GAN seeking to estimate the spoof traces. However, they formulate the traces as low-intensity noises, which is limited to print and replay attacks only and cannot provide convincing visual results. In contrast, we explore spoof traces for a much wider range of spoof attacks, visualize them with novel disentanglement, and also evaluate the proposed method on the challenging cases, *e.g.*, zero-shot face anti-spoofing.

Disentanglement Learning Disentanglement learning is often adopted to better represent complex data and features. DR-GAN Tran et al. (2017b) disentangles a face into identity and pose vectors for pose-invariant face recognition and view synthesis. Similarly in gait recognition, Zhang et al. (2019) disentangles the representations of appearance, canonical, and pose features from an input gait video. 3D reconstruction works Liu et al. (2018a); Tran & Liu (2021) also disentangle the representation of a 3D face into identity, expressions, poses, albedo, and illuminations. For image synthesis, Esser et al. (2018) disentangles an image into appearance and shape with U-Net and Variational Auto Encoder (VAE).

Different from Liu et al. (2018a); Tran et al. (2017b); Zhang et al. (2019), we intend to disentangle features that have different scales and contain geometric information. We leverage the multiple outputs to represent features at different scales, and adopt multiple-scale discriminators to properly learn them. Moreover, we propose a novel warping layer to tackle the geometric

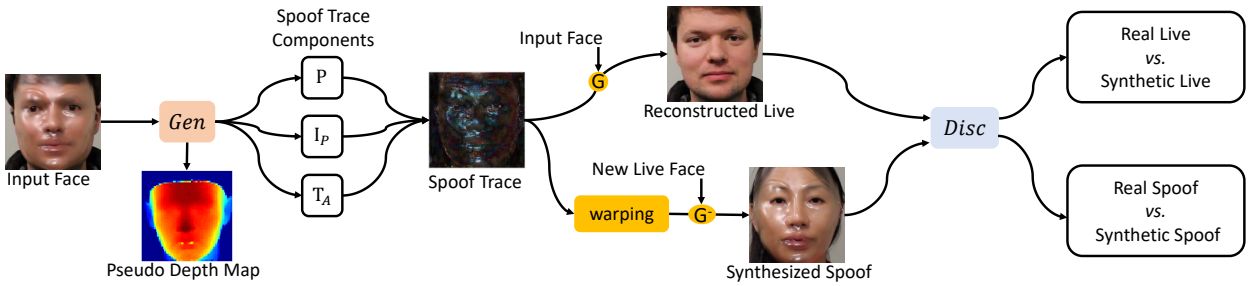


Figure 4.3 Overview of the proposed Physics-guided Spoof Trace Disentanglement (PhySTD).

discrepancy during the disentanglement and reconstruction.

Image Trace Modeling Image traces are certain signals existing in the image that can reveal information about the capturing camera, imaging setting, environment, and so on. Those signals often have much lower energy compared to the image content, which needs proper modeling to explore them. Abdelhamed et al. (2018); Thai et al. (2013, 2016) observe the difference of image noises, and use them to recognize the capture cameras. From the frequency domain, Stehouwer et al. (2020) shows the image noises from different cameras obey different noise distributions. Such techniques are applied to the field of image forensics, and later Chen et al. (2020); Wang et al. (2017) propose methods to remove such traces for image anti-forensics.

Recently, image trace modeling is widely used in image forgery detection and image adversarial attack detection Dang et al. (2020); Wu et al. (2019). In this work, we attempt to explore the traces of spoof face presentation. Due to different spoof mediums, spoof traces show large variations in content, intensity, and frequency distribution. We propose to disentangle the traces as additive traces and inpainting trace. And for additive traces, we further decompose them based on different frequency bands.

4.3 Physics-based Spoof Trace Disentanglement

4.3.1 Problem Formulation

Let the domain of live faces be denoted as $\mathcal{L} \subset \mathbb{R}^{N \times N \times 3}$ and spoof faces as $\mathcal{S} \subset \mathbb{R}^{N \times N \times 3}$, where N is the image size. We intend to obtain not only the correct prediction (live *vs.* spoof) of the input face, but also a convincing estimation of the spoof trace and live face reconstruction. To represent the spoof trace, our preliminary version assumes an additive relation between live and spoof, and uses 4 trace components $\{\mathbf{S}, \mathbf{B}, \mathbf{C}, \mathbf{T}\}$ at different frequency bands as:

$$\mathbf{I}_{spoof} = (1 + \lfloor \mathbf{S} \rfloor_{n_1}) \mathbf{I}_{live} + \lfloor \mathbf{B} \rfloor_{n_1} + \lfloor \mathbf{C} \rfloor_{n_2} + \mathbf{T}, \tag{4.1}$$

where $\mathbf{S}, \mathbf{B}$ represent low-frequency traces, $\mathbf{C}$ represents mid-frequency ones, and $\mathbf{T}$ represents high-frequency ones. $\lfloor \cdot \rfloor$ is the low bandpass filtering operation, and in practice, we achieve this by downsampling the original image and upsampling it back. In the previous setting, $n_1 = 1$ and $n_2 = 64$. Compared to the simple representation with only a single component in Jourabloo et al. (2018), this multi-scale representation of $\{\mathbf{S}, \mathbf{B}, \mathbf{C}, \mathbf{T}\}$ can largely improve disentanglement quality and suppress undesired artifacts due to its coarse-to-fine process. The model is designed to provide a valid estimation of spoof traces $\{\mathbf{S}, \mathbf{B}, \mathbf{C}, \mathbf{T}\}$ without respective ground truth. Our preliminary version Liu et al. (2020) aims to find a minimum intensity change that transfers an input face to the live domain:

$$\arg\min_{\hat{\mathbf{I}}} \|\mathbf{I} - \hat{\mathbf{I}}\|_F \; s.t. \; \mathbf{I} \in (\mathcal{S} \cup \mathcal{L}) \text{ and } \hat{\mathbf{I}} \in \mathcal{L}, \tag{4.2}$$

where $\mathbf{I}$ is the source face, $\hat{\mathbf{I}}$ is the target face to be optimized, and $\mathbf{I} - \hat{\mathbf{I}}$ is defined as the spoof trace. When the source face is live $\mathbf{I}_{\text{live}}$, $\mathbf{I} - \hat{\mathbf{I}}$ should be 0 as $\mathbf{I}$ is already in $\mathcal{L}$. When the source face is spoof $\mathbf{I}_{\text{spoof}}$, $\mathbf{I} - \hat{\mathbf{I}}$ should be regularized to prevent unnecessary changes such as identity shift.

Despite the effectiveness of this representation, there are still two drawbacks: First, the spoof trace disentanglement is mainly formulated as an additive processing. The optimization of Eqn. 4.2 limits the trace intensity, and the reconstruction for spoof regions with large appearance divergence might be sub-optimal, such as spoof glasses or mask. For those spoof regions, the physical relationship between the live and the spoof is better described as replacement rather than addition; Second, while our preliminary version representing the traces with hierarchical components, these components are learned with losses on their summation. Without careful supervision, the learned components can be ambiguous in their semantic meanings, *e.g.*, the high-frequency component may include low-frequency information.

To address the first drawback, we introduce a spoof generation function (SGF) as an additive process followed by an inpainting process:

$$\mathbf{I}_{spoof} = (1 - \mathbf{P})(\mathbf{I}_{live} + \mathbf{T}_A) + \mathbf{P} \cdot \mathbf{T}_P, \tag{4.3}$$

where $\mathbf{T}_A \in \mathbb{R}^{N \times N \times 3}$ indicates the traces from additive process, $\mathbf{T}_P$ indicates the traces from inpainting process, and $\mathbf{P} \in \mathbb{R}^{N \times N \times 1}$ denotes the inpainting region. Given a spoof face, one may reconstruct the live counterpart by inversing Eqn. 4.3:

$$\hat{\mathbf{I}}_{live} = (1 - \mathbf{P})(\mathbf{I}_{spoof} - \mathbf{T}_{\mathrm{A}}) + \mathbf{P} \cdot (\hat{\mathbf{I}}_{live} + \mathbf{I}_{spoof} - \mathbf{T}_P), \tag{4.4}$$

As the inpainting physically replaces content, the spoof trace $\mathbf{T}_P$ in the inpainting region $\mathbf{P}$ is identical to the spoof image $\mathbf{I}_{spoof}$ in the same region, and thus both cancel out in the second term of Eqn. 4.4. We further rename the $\hat{\mathbf{I}}_{live}$ in the second term as $\mathbf{I}_P$ to indicate the inpainting content within the inpainting region that should be estimated from the model. Therefore, the reconstruction

of the live image becomes:

$$\hat{\mathbf{I}}_{\textit{live}} = (1 - \mathbf{P})(\mathbf{I}_{\textit{spoof}} - \mathbf{T}_{\mathrm{A}}) + \mathbf{P} \cdot \mathbf{I}_P, \tag{4.5}$$

where $\mathbf{T}_A = \lfloor \mathbf{B} \rfloor_{n_1} + \lfloor \mathbf{C} \rfloor_{n_2} + \mathbf{T}$ denotes the additive trace represented by three hierarchical components. n_1 and n_2 are set to be 32 and 128 respectively. With a larger n_1, the effect of component $\mathbf{S}$ in the preliminary version can be incorporated into $\mathbf{B}$, and hence we remove $\mathbf{S}$ for simplicity. Besides the additive traces, the model is further required to estimate the inpainting region $\mathbf{P}$ and inpainting live content $\mathbf{I}_P$. $\mathbf{I}_P$ is estimated based on the rest of the live facial region without intensity constraint. We use a function $G(\cdot)$ to represent the reconstruction process of Eqn. 4.5. Accordingly, the optimization of Eqn. 4.2 is re-formulated by replacing $\hat{\mathbf{I}}$ with Eqn. 4.5 as:

$$\begin{aligned} &\underset{\mathbf{T}_A, \mathbf{P}, \mathbf{I}_P}{\arg\min} \ \|\mathbf{I} - (1 - \mathbf{P})(\mathbf{I} - \mathbf{T}_A) - \mathbf{P} \cdot \mathbf{I}_P\|_F \\ \rightarrow &\underset{\mathbf{T}_A, \mathbf{P}, \mathbf{I}_P}{\arg\min} \ \|(1 - \mathbf{P})\mathbf{T}_A\|_F + \|\mathbf{P} \cdot (\mathbf{I} - \mathbf{I}_P)\|_F. \end{aligned} \tag{4.6}$$

As we do not wish to impose any intensity constraint on $\mathbf{I}_P$, the final objective is formulated as:

$$\underset{\mathbf{T}_A, \mathbf{P}}{\arg\min} \|(1-\mathbf{P})\mathbf{T}_A\|_F + \lambda \|\mathbf{P}\|_F \ s.t. \ \mathbf{I} \in \mathcal{S} \cup \mathcal{L}, \hat{\mathbf{I}} \in \mathcal{L}, \tag{4.7}$$

where λ is a weight to balance two terms. In addition, based on Eqn. 4.3, we can define another function $G^{-}(\cdot)$ to synthesize new spoof faces, by transferring the spoof traces from $\mathbf{I}^i$ to $\mathbf{I}^j$:

$$\hat{\mathbf{I}}^{i \rightarrow j}_{\textit{spoof}} = G^{-}(\mathbf{I}^j | \mathbf{I}^i) = (1 - \mathbf{P}^i)(\mathbf{I}^j + \mathbf{T}^i_{\mathrm{A}}) + \mathbf{P}^i \cdot \mathbf{I}^i. \tag{4.8}$$

Note that $\mathbf{T}_P$ in Eqn. 4.3 has been replaced with $\mathbf{I}^i$ since the spoof image $\mathbf{I}^i$ contains the spoof

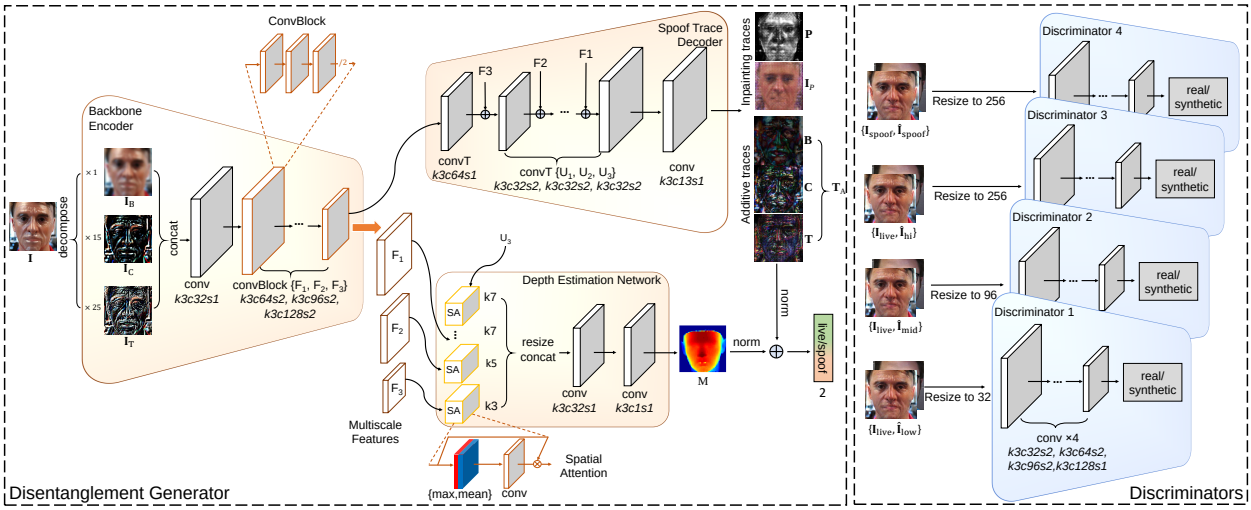


Figure 4.4 The proposed PhySTD network architecture. Except the last layer, each conv and transposed conv is concatenated with a batch normalizstion layer and a leaky ReLU layer. *k3c64s2* indicates the kernel size of 3×3, the convolution channel of 64 and the stride of 2.

trace for the inpainting region.

Estimating $\{\mathbf{T}_\mathrm{A}, \mathbf{P}, \mathbf{I}_P\}$ from an input face $\mathbf{I}$ is termed as **spoof trace disentanglement**. Given that no ground truth of traces is available, this disentanglement can be achieved via generative adversarial based training. As shown in Fig. 4.3, the proposed Physics-guided Spoof Trace Disentanglement (PhySTD) consists of a generator and discriminator. Given an input image, the generator is designed to predict the spoofness (represented by the pseudo depth map) as well as estimate the additive traces $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ and the inpainting components $\{\mathbf{P}, \mathbf{I}_P\}$. With the traces, we can apply function $G(\cdot)$ to reconstruct the live counterpart and function $G^{-}(\cdot)$ to synthesize new spoof faces. We adopt a set of discriminators at multiple image resolutions to distinguish the real faces $\{\mathbf{I}_{live}, \mathbf{I}_{spoof}\}$ with the synthetic faces $\{\hat{\mathbf{I}}_{live}, \hat{\mathbf{I}}_{spoof}\}$. To remedy the semantic ambiguity during $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ learning, three trace component combinations, $\{\mathbf{B}\}$, $\{\mathbf{B}, \mathbf{C}\}$, and $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$, will contribute to the synthesis of live reconstruction at one particular resolution, which is then supervised by a respective discriminator (details in Sec.4.3.3). To learn a proper inpainting region $\mathbf{P}$, we leverage both the prior knowledge and the information from the additive traces.

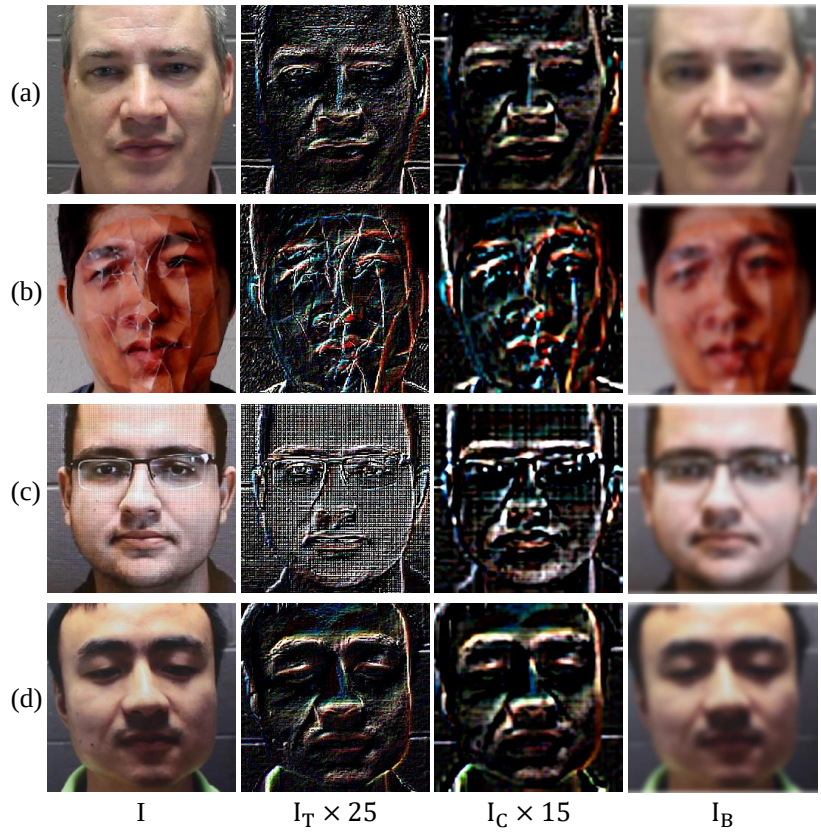


Figure 4.5 The visualization of image decomposition for different input faces: (a) live face (b) 3D mask attack (c) replay attack (d) print attack.

In the rest of this section, we present the details of the generator, the discriminators, the details of face reconstruction and synthesis, and the losses and training steps used in PhySTD.

4.3.2 Disentanglement Generator

As shown in Fig. 4.4, the disentanglement generator consists of a backbone encoder, a spoof trace decoder and a depth estimation network. The backbone encoder aims to extract multi-scale features, the depth estimation network leverages the features to estimate the facial depth map, and a spoof trace decoder to estimate the additive trace components $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ and the inpainting components $\{\mathbf{P}, \mathbf{I}_P\}$. The depth map and the spoof traces will be used to compute the final spoofness score.

Backbone encoder Backbone encoder extracts features from the input images for both depth map

estimation and spoof trace disentanglement. As shown in our preliminary work Liu et al. (2020), the spoof traces consists of components from different frequency bands: low-frequency traces includes color distortion, mid-frequency traces includes makeup strikes, and high-frequency traces includes Moir patterns and mask edges. However, a vanilla CNN model might overlook high-frequency traces since the energy of high-frequency traces is often much weaker than that of low-frequency traces. In order to encourage the network to equally regard traces with different physical properties, we explicitly decompose the image into three elements $\{\mathbf{I_B}, \mathbf{I_C}, \mathbf{I_T}\}$ as:

$$\begin{aligned} \mathbf{I_B} &= \lfloor \mathbf{I} \rfloor_{n_1}, \\ \mathbf{I_C} &= \lfloor \mathbf{I} \rfloor_{n_2} - \lfloor \mathbf{I} \rfloor_{n_1}, \\ \mathbf{I_T} &= \mathbf{I} - \lfloor \mathbf{I} \rfloor_{n2}, \end{aligned} \tag{4.9}$$

where $n_1 = 32$, $n_2 = 128$ and the image size $N = 256$. In addition, we amplify the value in $\mathbf{I_C}, \mathbf{I_T}$ by two constants 15 and 25, and then feed the concatenation of three elements to the backbone network. Fig. 4.5 provides the visualization of image decomposition. We observe that the traces that are less distinct in the original images become more highlighted in the $\mathbf{I_T}$ component: 3D mask and replay attack bring unique patterns different with the live face pattern, while print attack is lacking of necessary high frequency details. Semantically, $\mathbf{I_B}, \mathbf{I_C}, \mathbf{I_T}$ share the same frequency domains with $\mathbf{B}, \mathbf{C}, \mathbf{T}$ respectively, and thus the decomposition potentially eases the learning of $\mathbf{B}, \mathbf{C}, \mathbf{T}$.

After that, the encoder progressively downsamples the decomposed image components 3 times to obtain features $\mathbf{F}_1 \in \mathbb{R}^{128\times128\times64}$, $\mathbf{F}_2 \in \mathbb{R}^{64\times64\times96}$, $\mathbf{F}_3 \in \mathbb{R}^{32\times32\times128}$ via conv layers.

Spoof trace decoder The decoder upsamples the feature $\mathbf{F}_3$ with transpose conv layers back to the input face size 256. The last layer outputs both additive traces $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ and inpainting components $\{\mathbf{P}, \mathbf{I}_P\}$. Similar to U-Net Ronneberger et al. (2015), we apply the short-cut connection

between the backbone encoder and decoder to bypass the multiple scale details for a high-quality trace estimation.

Depth estimation network We still recognize the importance of the discriminative supervision used in auxiliary FAS, and thus introduce a depth estimation network to perform the pseudo-depth estimation for face anti-spoofing, as proposed in Liu et al. (2018c). The depth estimation network takes the concatenated features of $\mathbf{F}_1$, $\mathbf{F}_2$, $\mathbf{F}_3$ from the backbone encoder and $\mathbf{U}_3$ from the decoder as input. The features are put through a spatial attention mechanism from Yu et al. (2020b) and resize to the same size of $K = 32$. It outputs a face depth map $\mathbf{M} \in \mathbb{R}^{32\times32}$, where the depth values are normalized within $[0, 1]$. Regarding the number of parameters, both spoof trace decoder and depth estimation network are light weighed, while the backbone network is much heavier. With more network layers being shared to tackle both depth estimation and spoof trace disentanglement, the knowledge learnt from spoof trace disentanglement can be better shared with depth estimation task, which can lead to a better anti-spoofing performance.

Final scoring In the testing phase, we use the norm of the depth map and the intensity of spoof traces for real vs. spoof classification:

$$\text{score} = \frac{1}{2K^2}\|\mathbf{M}\|_1 + \frac{\alpha_0}{2N^2}(\|\mathbf{B}\|_1 + \|\mathbf{C}\|_1 + \|\mathbf{T}\|_1 + \|\mathbf{P}\|_1), \tag{4.10}$$

where α_0 is the weight for the spoof trace.

4.3.3 Reconstruction and Synthesis

There are multiple options to use the disentangled spoof traces: 1) live reconstruction, 2) spoof synthesis, and 3) "harder" sample synthesis, which will be described below respectively.

Live reconstruction: Based on Eqn. 4.5, we propose a hierarchical reconstruction of the live face

counterpart from the input images. To reconstruct faces at a certain resolution, each additive trace is included only if its frequency domain is lower than the target resolution. We apply $\{hi, mid, low\}$ three resolution settings as:

$$\begin{aligned}
\hat{\mathbf{I}}_{hi} =& (1-\mathbf{P})(\mathbf{I} - \lfloor \mathbf{B} \rfloor_{n_1} - \lfloor \mathbf{C} \rfloor_{n_2} - \mathbf{T}) + \mathbf{P} \cdot \mathbf{I}_P, \\
\hat{\mathbf{I}}_{mid} =& (1-\mathbf{P})(\lfloor \mathbf{I} \rfloor_{n_2} - \lfloor \mathbf{B} \rfloor_{n_1} - \lfloor \mathbf{C} \rfloor_{n_2}) + \mathbf{P} \cdot \mathbf{I}_P, \\
\hat{\mathbf{I}}_{low} =& (1-\mathbf{P})(\lfloor \mathbf{I} \rfloor_{n_1} - \lfloor \mathbf{B} \rfloor_{n_1}) + \mathbf{P} \cdot \mathbf{I}_P.
\end{aligned} \tag{4.11}$$

Spoof synthesis: Based on Eqn. 4.8, we can obtain a new spoof face via applying the spoof traces disentangled from a spoof face $\mathbf{I}_i$ to a live face $\mathbf{I}_j$. However, spoof traces may contain face-dependent content associated with the original spoof subject. Directly applying them to a new face with different shapes or poses may result in mis-alignment and strong visual implausibility. Therefore, the spoof trace should go through a geometry correction before performing this synthesis. We propose an online 3D warping layer and will introduce it in the following subsection.

"Harder" sample synthesis: The disentangled spoof traces can not only reconstruct live and synthesize new spoof, but also synthesize "harder" spoof samples by removing or amplifying part of the spoof traces. We can tune one or some of the trace elements $\{\mathbf{B}, \mathbf{C}, \mathbf{T}, \mathbf{P}\}$ to make the spoof sample to become "less spoofed", which is thus closer to a live face since the spoof traces are weakened. Such spoof data can be regarded as *harder* samples and may benefit the generalization of the disentanglement generator. For instance, while removing the low frequency element $\mathbf{B}$ from a replay spoof trace, the generator may be forced to rely on other elements such as high-level texture patterns. To synthesize the "harder" sample $\hat{\mathbf{I}}_{hard}$, we follow Eqn. 4.8 with two minor changes: 1) generate 3 random weights between $[0, 1]$ and multiple each with one component of $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$; 2) randomly remove the inpainting process (*i.e.*, set $\mathbf{P} = 0$) with a probability of 0.5. Compared with

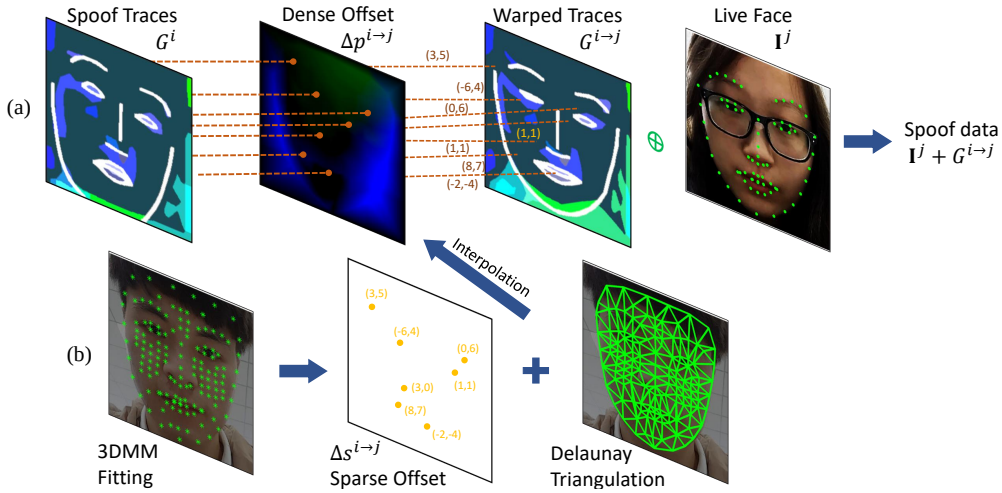


Figure 4.6 The online 3D warping layer. (a) Given the corresponding dense offset, we warp the spoof trace and add them to the target live face to create a new spoof. E.g. pixel (x, y) with offset $(3, 5)$ is warped to pixel$(x + 3, y + 5)$ in the new image. (b) To obtain a dense offsets from the spare offsets of the selected face shape vertices, Delaunay triangulation interpolation is adopted.

other methods, such as brightness and contrast change Liu et al. (2019b), reflection and blurriness effect Yang et al. (2019a), or 3D distortion Guo et al. (2019), our approach can introduce more realistic and effective data samples, as shown in Sec. 4.4.

4.3.3.1 Online 3D Warping Layer

We propose an online 3D warping layer to correct the shape discrepancy. To obtain the warping, previous methods in Chang et al. (2018); Liu et al. (2018c) use offline face swapping and pre-computed dense offset respectively, where both methods are non-differentiable as well as memory intensive. In contrast, our warping layer is designed to be both differentiable and computationally efficient, which is necessary for online synthesis during the training.

First, the live reconstruction of a spoof face $\mathbf{I}^i$ can be expressed as:

$$G^i = G(\mathbf{I}^i)[\mathbf{p}_0], \tag{4.12}$$

where $\mathbf{p}^0 = \{(0,0),(0,1),...,(255,255)\} \in \mathbb{R}^{256\times256\times2}$ enumerates pixel locations in $\mathbf{I}^i$. To align the spoof traces while synthesizing a new spoof face, a dense offset $\Delta\mathbf{p}^{i\to j} \in \mathbb{R}^{256\times256\times2}$ is required to indicate the deformation between face $\mathbf{I}^i$ and face $\mathbf{I}^j$. A discrete deformation can be acquired from the distances of the corresponding facial landmarks between two faces. During the data preparation, we use Liu et al. (2017) to fit a 3DMM model and extract the 2D locations of Q facial vertices for each face:

$$\mathbf{s} = \{(x_0,y_0),(x_1,y_1),...,(x_N,y_N)\} \in \mathbb{R}^{Q\times2}. \tag{4.13}$$

A sparse offset on the corresponding vertices can then be computed two faces as $\Delta\mathbf{s}^{i\to j} = \mathbf{s}^j - \mathbf{s}^i$. To convert the sparse offset $\Delta\mathbf{s}^{i\to j}$ to the dense offset $\Delta\mathbf{p}^{i\to j}$, we apply a triangulation interpolation:

$$\Delta\mathbf{p}^{i\to j} = \text{Tri}(\mathbf{p}^0, \mathbf{s}^i, \Delta\mathbf{s}^{i\to j}), \tag{4.14}$$

where $\text{Tri}(\cdot)$ is the interpolation, $\mathbf{s}^i$ denotes the vertex locations, $\Delta\mathbf{s}^{i\to j}$ are the vertex values, and we adopt Delaunay triangulation. The warping operation can be denoted as:

$$G^{-i\to j} = G^-(\mathbf{I}^j|\mathbf{I}^i)[\mathbf{p}^0 + \Delta\mathbf{p}^{i\to j}], \tag{4.15}$$

where the offset $\Delta\mathbf{p}^{i\to j}$ applies to all subject i related elements $\{\mathbf{T}^i_A, \mathbf{I}^i, \mathbf{P}^i\}$. Since the offset $\Delta\mathbf{p}^{i\to j}$ is typically composed of fractional numbers, we implement the bilinear interpolation to sample the fractional pixel locations. We select $Q = 140$ vertices to cover the face region so that they can represent non-rigid deformation, due to pose and expression. As the pixel values in the warped face are a linear combination of pixel values of the triangulation vertices, this entire process is differentiable. This process is illustrated in Fig. 4.6.

Algorithm 1 PhySTD Training Iteration.

Input: live faces $\mathbf{I}_{live}$ and facial landmarks $\mathbf{s}_{live}$, spoof faces $\mathbf{I}_{spoof}$ and facial landmarks $\mathbf{s}_{spoof}$, ground truth depth map $\mathbf{M}_0$, preliminary mask $\mathbf{P}_0$;

Output: reconstructed live $\hat{\mathbf{I}}_{live}$, synthesized spoof $\hat{\mathbf{I}}_{spoof}$, spoof traces $\{\mathbf{T}_A^l, \mathbf{P}^l, \mathbf{I}_P^l, \mathbf{T}_A^s, \mathbf{P}^s, \mathbf{I}_P^s\}$, depth maps $\{\mathbf{M}^l, \mathbf{M}^s\}$;

while *iteration* $<$ *max_iteration* **do**

`// training step 1`

1: compute $\mathbf{T}_A^l, \mathbf{P}^l, \mathbf{I}_P^l \leftarrow G(\mathbf{I}_{live})$ and compute $\mathbf{T}_A^s, \mathbf{P}^s, \mathbf{I}_P^s \leftarrow G(\mathbf{I}_{spoof})$;

2: estimate the depth map $\mathbf{M}^l$, $\mathbf{M}^s$;

3: compute losses L_{depth}, L_P, L_R;

`// training step 2`

4: compute $\hat{\mathbf{I}}_{low}$, $\hat{\mathbf{I}}_{mid}$, $\hat{\mathbf{I}}_{hi}$ from $\mathbf{T}_A^s$, $\mathbf{P}^s$, $\mathbf{I}_P^s$ and $\mathbf{I}_{spoof}$ (Eqn. 4.4);

5: compute warping offset $\Delta\mathbf{p}^{s\to l}$ from $\mathbf{s}_{live}$, $\mathbf{s}_{spoof}$ (Eqn. 5.8);

6: compute $\hat{\mathbf{I}}_{spoof}$ from warped $\mathbf{T}_A^{s\to l}$, $\mathbf{P}^{s\to l}$ and $\mathbf{I}_{live}$ (Eqn. 5.9);

7: send $\mathbf{I}_{live}$, $\mathbf{I}_{spoof}$, $\hat{\mathbf{I}}_{low}$, $\hat{\mathbf{I}}_{mid}$, $\hat{\mathbf{I}}_{hi}$, $\hat{\mathbf{I}}_{spoof}$ to discriminators;

8: compute the adversarial loss for generator L_G and for discriminators L_D;

`// training step 3`

9: create harder samples $\mathbf{I}_{hard}$ from $\mathbf{T}_A^{s\to l}$, $\mathbf{P}^{s\to l}$ and $\mathbf{I}_{live}$ with random perturbation on traces;

10: compute $\mathbf{T}_A^h, \mathbf{P}^h, \mathbf{I}_P^h \leftarrow G(\mathbf{I}_{hard})$;

11: compute depth map $\mathbf{M}^h$ for $\mathbf{I}_{hard}$;

12: compute losses L_S, L_H;

`// back propagation`

13: back-propagate the losses from step $3, 8, 12$ to corresponding parts and update the network;

end

4.3.4 Multi-scale Discriminators

Motivated by Wang et al. (2018b), we adopt multiple discriminators at different resolutions (*e.g.*, 32, 96, and 256) in our GAN architecture. We follow the design of PatchGAN Isola et al. (2017), which essentially is a fully convolutional network. Fully convolutional networks are shown to be effective to not only synthesize high-quality images Isola et al. (2017); Wang et al. (2018b), but also tackle face anti-spoofing problems Liu et al. (2018c). For each discriminator, we adopt the same structure but do not share the weights.

As shown in Fig. 4.4, we use in total 4 discriminators in our work: D_1, working in the lowest resolution of 32, focuses on low frequency elements since the higher-frequency traces are erased by downsampling. D_2, working at the resolution of 96, focuses on the middle level content pattern. D_3 and D_4, working on the highest resolution of 256, focus on the fine texture details. Our preliminary version resizes real and synthetic samples $\{\mathbf{I}, \hat{\mathbf{I}}\}$ to different resolutions and assign to each discriminator. To remove semantic ambiguity and provide correspondence to the trace components, we instead assign the hierarchical reconstruction from Eqn. 4.11 to the discriminators: we send low frequency pairs $\{\mathbf{I}_{\textit{live}}, \hat{\mathbf{I}}_{\textit{low}}\}$ to D_1, middle frequency pairs $\{\mathbf{I}_{\textit{live}}, \hat{\mathbf{I}}_{\textit{mid}}\}$ to D_2, high frequency pairs $\{\mathbf{I}_{\textit{live}}, \hat{\mathbf{I}}_{\textit{hi}}\}$ to D_3, and real/synthetic spoof $\{\mathbf{I}_{\textit{spoof}}, \hat{\mathbf{I}}_{\textit{spoof}}\}$ to D_4. Each discriminator outputs a 1-channel map in the range of $[0, 1]$, where 0 denotes fake and 1 denotes real.

4.3.5 Loss Functions and Training Steps

We utilize multiple loss functions to supervise the learning of depth maps and spoof traces. Each training iteration consists of three training steps. We first introduce the loss function, followed by how they are used in the training steps.

Depth map loss: We follow the auxiliary FAS in Liu et al. (2018c) to estimate an auxiliary depth map $\mathbf{M}$, where the depth ground truth $\mathbf{M}_0$ for a live face contains face-like shape and the depth for spoof should be zero. We apply the $\mathcal{L}$-1 norm on this loss as:

$$L_{\textit{depth}} = \frac{1}{K^2}\mathbb{E}_{i\sim\mathcal{L}\cup\mathcal{S}}\|\mathbf{M}^i - \mathbf{M}_0^i\|_F, \tag{4.16}$$

where $K = 32$ is the size of $\mathbf{M}$. We apply the dense face alignment Liu et al. (2017) to estimate the 3D shape and render the depth ground truth $\mathbf{M}_0$.

Adversarial loss for G: We employ the LSGANs Mao et al. (2017) on reconstructed live faces

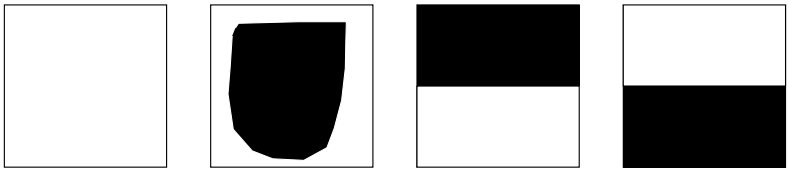


Figure 4.7 Preliminary mask $\mathbf{P}_0$ for the negative term in inpainting mask loss. White pixels denote 1 and black pixels denote 0. White indicates the area should not be inpainted. $\mathbf{P}_0$ for: (a) print, replay; (b) 3D mask and makeup; (c) partial attacks that cover the eye portion; (d) partial attacks that cover the mouth portion.

and synthesized spoof faces. It encourages the reconstructed live to look similar to real live from domain $\mathcal{L}$, and the synthesized spoof faces to look similar to faces from domain $\mathcal{S}$:

$$L_G = \mathbb{E}_{i\sim\mathcal{L},j\sim\mathcal{S}}\Big[\|D_1(\hat{\mathbf{I}}^j_{low})-\mathbf{1}\|_F^2 + \|D_2(\hat{\mathbf{I}}^j_{mid})-\mathbf{1}\|_F^2 + \|D_3(\hat{\mathbf{I}}^j_{hi})-\mathbf{1}\|_F^2 + \|D_4(\hat{\mathbf{I}}^{j\to i}_{spoof})-\mathbf{1}\|_F^2\Big]. \tag{4.17}$$

Adversarial loss for D: The adversarial loss for discriminators encourages $D(\cdot)$ to distinguish between real live *vs.* reconstructed live, and real spoof *vs.* synthesized spoof:

$$\begin{aligned} L_D = \mathbb{E}_{i\sim\mathcal{L},j\sim\mathcal{S}}\Big[&\|D_1(\mathbf{I}^i)-\mathbf{1}\|_F^2 + \|D_2(\mathbf{I}^i)-\mathbf{1}\|_F^2 + \|D_3(\mathbf{I}^i)-\mathbf{1}\|_F^2 + \|D_4(\mathbf{I}^j)-\mathbf{1}\|_F^2 + \\ &\|D_1(\hat{\mathbf{I}}^j_{low})\|_F^2 + \|D_2(\hat{\mathbf{I}}^j_{mid})\|_F^2 + \|D_3(\hat{\mathbf{I}}^j_{hi})\|_F^2 + \|D_4(\hat{\mathbf{I}}^{j\to i}_{spoof})\|_F^2\Big]. \end{aligned} \tag{4.18}$$

Inpainting mask loss: The ground truth inpainting region for all spoof attacks is barely possible to obtain, hence a fully supervised training used in Tran et al. (2017a) for inpainting mask is out of the question. However, we may still leverage the prior knowledge of spoof attacks to facilitate the estimation of inpainting masks. The inpainting mask loss consists of a positive term and a negative term. First, the positive term encourages certain region to be inpainted. As the goal of inpainting process is to allow certain region to change without intensity constraint, the region with larger magnitude of additive traces would have a higher probability to be inpainted. Hence, the positive term adopts a $\mathcal{L}$-2 norm between the inpainting region $\mathbf{P}$ and the region where the additive trace is

larger than a threshold β.

Second, the negative term discourages certain region to be inpainted. While the ground truth inpainting mask is unknown, it's straightforward to mark a large portion of region that should not be inpainted. For instance, the inpainting region for funny eye glasses should not appear in the lower part of a face. Hence, we provide a preliminary mask $\mathbf{P}_0$ to indicate the not-to-be-inpainted region, and adopt a normalized $\mathcal{L}$-2 norm on the masked inpainting region $\mathbf{P} \cdot \mathbf{P}_0$ as the negative term. The preliminary mask $\mathbf{P}_0$ is illustrated in Fig. 4.7. Overall, the inpainting mask loss is formed as:

$$L_P = \mathbb{E}_{\mathbf{i}\sim\mathcal{S}}\Big[\|\mathbf{P}^i - (\mathbf{T}_A^i > \beta)\|_F^2 + \frac{\|\mathbf{P}^i \cdot \mathbf{P}_0^i\|_F^2}{\|\mathbf{P}_0^i\|_F^2}\Big]. \tag{4.19}$$

Trace regularization: Based on Eqn. 4.6 with $\lambda = 1$, we regularize the intensity of additive traces $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ and inpainting region $\mathbf{P}$. The regularizer loss is denoted as:

$$L_R = \mathbb{E}_{\mathbf{i}\sim\mathcal{L}\cup\mathcal{S}}\Big[\|\mathbf{B}\|_F^2 + \|\mathbf{C}\|_F^2 + \|\mathbf{T}\|_F^2 + \|\mathbf{P}\|_F^2\Big]. \tag{4.20}$$

Synthesized spoof loss: Synthesized spoof data come with ground truth spoof traces. As a result, we are able to define a supervised pixel loss for the generator to disentangle the exact spoof traces that were added:

$$L_S = \mathbb{E}_{\mathbf{i}\sim\mathcal{L},\mathbf{j}\sim\mathcal{S}}\Big[\|G(\lceil G^{-j\to i}\rceil) - \lceil G^{j\to i}\rceil\|_F^1\Big], \tag{4.21}$$

where $G^{j\to i}$ is the overall effect of $\{\mathbf{P}^j, \mathbf{I}_P^j, \mathbf{B}^j, \mathbf{C}^j, \mathbf{T}^j\}$ after warping to subject i, and $\lceil\cdot\rceil$ is the `stop_gradient` operation. Without stopping the gradient, $G^{j\to i}$ may collapse to 0.

Depth map loss for "harder" samples: We send the "harder" synthesized spoof data to depth estimation network to improve the data diversity, and hope to increase the FAS model's generaliza-

tion:

$$L_H = \frac{1}{K^2}\mathbb{E}_{i\sim\hat{\mathcal{S}}}\Big[\|\mathbf{M}^i - \mathbf{M}_0^i\|_F\Big], \tag{4.22}$$

where $\hat{\mathcal{S}}$ denotes the domain of synthesized spoof faces.

Training steps and total loss: Each training iteration has 3 training steps. In the training step 1, live faces $\mathbf{I}_{live}$ and spoof faces $\mathbf{I}_{spoof}$ are fed into generator $G(\cdot)$ to disentangle the spoof traces. The spoof traces are used to reconstruct the live counterpart $\hat{\mathbf{I}}_{live}$ and synthesize new spoof $\hat{\mathbf{I}}_{spoof}$. The generator is updated with respect to the depth map loss L_{depth}, adversarial loss L_G, inpainting mask loss L_P, and regularizer loss L_R:

$$L = \alpha_1 L_{depth} + \alpha_2 L_G + \alpha_3 L_P + \alpha_4 L_R. \tag{4.23}$$

In the training step 2, the discriminators are supervised with the adversarial loss L_D to compete with the generator. In the training step 3, $\mathbf{I}_{live}$ and $\hat{\mathbf{I}}_{hard}$ are fed into the generator with the ground truth label and trace to minimize the synthesized spoof loss L_S and depth map loss L_H:

$$L = \alpha_5 L_S + \alpha_6 L_H, \tag{4.24}$$

where α_1-α_6 are the weights to balance the multitask training. To note that, we send the original live faces $\mathbf{I}_{live}$ with $\hat{\mathbf{I}}_{hard}$ for a balanced mini-batch, which is important when computing the moving average in the batch normalization layer. We execute all 3 steps in each minibatch iteration, but reduce the learning rate for discriminator step by half. The whole training process is depicted in Alg. 1.

Protocol	Method	APCER (%)	BPCER (%)	ACER (%)
1	STASN (Yang et al. (2019a))	1.2	2.5	1.9
	Auxiliary (Liu et al. (2018c))	1.6	1.6	1.6
	DeSpoof (Jourabloo et al. (2018))	1.2	1.7	1.5
	DRL (Zhang et al. (2020a))	1.7	0.8	1.3
	STDN (Liu et al. (2020))	0.8	1.3	1.1
	CDCN (Yu et al. (2020b))	0.4	1.7	1.0
	HMP (Yu et al. (2020a))	0.0	1.6	0.8
	CDCN++ (Yu et al. (2020b))	0.4	**0.0**	**0.2**
	Ours	**0.0**	0.8	0.4
2	DeSpoof (Jourabloo et al. (2018))	4.2	4.4	4.3
	Auxiliary (Liu et al. (2018c))	2.7	2.7	2.7
	DRL (Zhang et al. (2020a))	1.1	3.6	2.4
	STASN (Yang et al. (2019a))	4.2	**0.3**	2.2
	STDN (Liu et al. (2020))	2.3	1.6	1.9
	HMP (Yu et al. (2020a))	2.6	0.8	1.7
	CDCN (Yu et al. (2020b))	0.4	1.7	1.5
	CDCN++ (Yu et al. (2020b))	1.8	0.8	1.3
	Ours	**1.2**	1.3	**1.3**
3	DeSpoof (Jourabloo et al. (2018))	4.0 ± 1.8	3.8 ± 1.2	3.6 ± 1.6
	Auxiliary (Liu et al. (2018c))	2.7 ± 1.3	3.1 ± 1.7	2.9 ± 1.5
	STDN (Liu et al. (2020))	1.6 ± 1.6	4.0 ± 5.4	2.8 ± 3.3
	STASN (Yang et al. (2019a))	4.7 ± 3.9	$\mathbf{0.9 \pm 1.2}$	2.8 ± 1.6
	HMP (Yu et al. (2020a))	2.8 ± 2.4	2.3 ± 2.8	2.5 ± 1.1
	CDCN (Yu et al. (2020b))	2.4 ± 1.3	2.2 ± 2.0	2.3 ± 1.4
	DRL (Zhang et al. (2020a))	2.8 ± 2.2	1.7 ± 2.6	2.2 ± 2.2
	CDCN++ (Yu et al. (2020b))	1.7 ± 1.5	2.0 ± 1.2	$\mathbf{1.8 \pm 0.7}$
	Ours	$\mathbf{1.7 \pm 1.4}$	2.2 ± 3.5	1.9 ± 2.3
4	Auxiliary (Liu et al. (2018c))	9.3 ± 5.6	10.4 ± 6.0	9.5 ± 6.0
	STASN (Yang et al. (2019a))	6.7 ± 10.6	8.3 ± 8.4	7.5 ± 4.7
	CDCN (Yu et al. (2020b))	4.6 ± 4.6	9.2 ± 8.0	6.9 ± 2.9
	DeSpoof (Jourabloo et al. (2018))	5.1 ± 6.3	6.1 ± 5.1	5.6 ± 5.7
	HMP (Yu et al. (2020a))	2.9 ± 4.0	7.5 ± 6.9	5.2 ± 3.7
	CDCN++ (Yu et al. (2020b))	4.2 ± 3.4	5.8 ± 4.9	5.0 ± 2.9
	DRL (Zhang et al. (2020a))	5.4 ± 2.9	$\mathbf{3.3 \pm 6.0}$	4.8 ± 6.4
	STDN (Liu et al. (2020))	2.3 ± 3.6	5.2 ± 5.4	3.8 ± 4.2
	Ours	$\mathbf{2.3 \pm 3.6}$	4.2 ± 5.4	$\mathbf{3.6 \pm 4.2}$

Table 4.1 The evaluation on four protocols in OULU-NPU. **Bold** indicates the best score in each protocol.

4.4 Experiments

In this section, we first introduce the experimental setup, and then present the results in the known, unknown, and open-set spoof scenarios, with comparisons to respective baselines. Next, we quantitatively evaluate the spoof traces by performing a spoof medium classification, and conduct an ablation study on each design in the proposed method. Finally, we provide visualization results on the spoof trace disentanglement, new spoof synthesis and t-SNE visualization.

4.4.1 Experimental Setup

Databases We conduct experiments on three major databases: Oulu-NPU Boulkenafet et al. (2017b), SiW Liu et al. (2018c), and SiW-M Liu et al. (2019a). Oulu-NPU and SiW include print/replay attacks, while SiW-M includes 13 spoof types. We follow all the existing testing protocols and compare with SOTA methods. Similar to most prior works, we only use the face region for training and testing.

Evaluation metrics Two common metrics are used in this work for comparison: EER and APCER/BPCER/ACER. EER describes the theoretical performance and predetermines the threshold for making decisions. APCER/BPCER/ACER in ISO/IEC-JTC-1/SC-37 (2016) describe the practical performance given a predetermined threshold. For both evaluation metrics, lower value means better performance. The threshold for APCER/BPCER/ACER is computed from either training set or validation set. In addition, we also report the True Detection Rate (TDR) at a given False Detection Rate (FDR). This metric describes the spoof detection rate at a strict tolerance to live errors, which is widely used to evaluate real-world systems IARPA (2016). In this work, we report TDR at FDR$= 0.5\%$. For TDR, the higher the better.

Parameter setting PhySTD is implemented in Tensorflow with an initial learning rate of $5e$-5. We train in total $150,000$ iterations with a batch size of 8, and decrease the learning rate by a ratio of 10 every $45,000$ iterations. We initialize the weights with $[0, 0.02]$ normal distribution. $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6\}$ are set to be $\{100, 5, 1, 1e\text{-}4, 10, 1\}$, and $\beta = 0.1$. α_0 is empirically determined from the training or validation set. We use the open-source face alignment Bulat & Tzimiropoulos (2017) and 3DMM fitting Liu et al. (2017) to crop the face and provide 140 landmarks.

Protocol	Method	APCER (%)	BPCER (%)	ACER (%)
1	Auxiliary (Liu et al. (2018c))	3.6	3.6	3.6
	STASN (Yang et al. (2019a))	–	–	1.0
	Meta-FAS-DR (Zhao et al. (2019))	0.5	0.5	0.5
	HMP (Yu et al. (2020a))	0.6	0.2	0.5
	DRL (Zhang et al. (2020a))	0.1	0.5	0.3
	CDCN (Yu et al. (2020b))	0.1	0.2	0.1
	CDCN++ (Yu et al. (2020b))	0.1	0.2	0.1
	Ours	**0.0**	**0.0**	**0.0**
2	Auxiliary (Liu et al. (2018c))	0.6 ± 0.7	0.6 ± 0.7	0.6 ± 0.7
	Meta-FAS-DR (Zhao et al. (2019))	0.3 ± 0.3	0.3 ± 0.3	0.3 ± 0.3
	STASN (Yang et al. (2019a))	–	–	0.3 ± 0.1
	HMP (Yu et al. (2020a))	0.1 ± 0.2	0.2 ± 0.0	0.1 ± 0.1
	DRL (Zhang et al. (2020a))	0.1 ± 0.2	0.1 ± 0.1	0.1 ± 0.0
	CDCN (Yu et al. (2020b))	0.0 ± 0.0	0.1 ± 0.1	0.1 ± 0.0
	CDCN++ (Yu et al. (2020b))	0.0 ± 0.0	0.1 ± 0.1	0.0 ± 0.1
	Ours	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$	$\mathbf{0.0 \pm 0.0}$
3	STASN (Yang et al. (2019a))	–	–	12.1 ± 1.5
	Auxiliary (Liu et al. (2018c))	8.3 ± 3.8	8.3 ± 3.8	8.3 ± 3.8
	Meta-FAS-DR (Zhao et al. (2019))	8.0 ± 5.0	7.4 ± 5.7	7.7 ± 5.3
	DRL (Zhang et al. (2020a))	9.4 ± 6.1	1.8 ± 2.6	5.6 ± 4.4
	HMP (Yu et al. (2020a))	2.6 ± 0.9	2.3 ± 0.5	2.5 ± 0.7
	CDCN (Yu et al. (2020b))	2.4 ± 1.3	2.2 ± 2.0	2.3 ± 1.4
	CDCN++ (Yu et al. (2020b))	$\mathbf{1.7 \pm 1.5}$	$\mathbf{2.0 \pm 1.2}$	$\mathbf{1.8 \pm 0.7}$
	Ours	13.1 ± 9.4	1.6 ± 0.6	7.4 ± 4.3

Table 4.2 The evaluation on three protocols in SiW Dataset. We compare with the top 7 performances.

4.4.2 Anti-Spoofing for Known Spoof Types

Oulu-NPU Oulu-NPUBoulkenafet et al. (2017b) is a commonly used face anti-spoofing benchmark due to its high-quality data and challenging testing protocols. Tab. 4.1 shows our anti-spoofing performance on Oulu-NPU, compared with SOTA algorithms. Our method achieves the best overall performance on this database. Compared with our preliminary version Liu et al. (2020), we demonstrate improvements in all 4 protocols, with significant improvement on protocol 1 and protocol 3, *i.e.*, reducing the ACER by 63.6% and 32.1% respectively. Compared with the SOTA, our approach achieves similar best performances on the first three protocols and outperforms the SOTA on the fourth protocol, which is the most challenging one. To note that, in protocol 3 and protocol 4, the performances of testing camera 6 are much lower than those of cameras 1-5: the ACER for camera 6 are 6.4% and 10.2%, while the average ACER for the other cameras are 1.0% and 2.0% respectively. Compared with other cameras, we notice that camera 6 has stronger sensor

Metrics(%)	Method	Replay	Print	3D Mask					Makeup			Partial Attacks			Overall
				Half	Silic.	Trans.	Paper	Mann.	Ob.	Im.	Cos.	Funny.	Papergls.	Paper	
ACER	Auxiliary (Liu et al. (2018c))	5.1	5.0	5.0	10.2	5.0	9.8	6.3	19.6	5.0	26.5	5.5	5.2	5.0	6.3
	SDTN (Liu et al. (2020))	3.2	3.1	3.0	9.0	3.0	3.4	4.7	**3.0**	3.0	24.5	4.1	3.7	3.0	4.1
	Step1	6.1	5.4	5.4	5.4	5.4	5.4	5.4	22.7	5.4	26.8	5.4	5.5	5.4	10.9
	Step1+Step2 w/ single trace	8.7	7.8	7.8	7.8	7.8	7.8	7.8	25.0	7.9	28.8	7.8	7.8	7.8	13.8
	Step1+Step2	4.1	3.9	3.9	3.9	4.0	3.9	4.0	13.5	4.0	25.1	3.9	3.9	3.9	4.6
	Step1+Step2+Step3 (Ours)	**3.2**	**1.4**	**1.0**	**2.3**	**1.3**	**2.9**	**2.5**	12.4	**1.2**	**18.5**	**1.7**	**0.4**	**1.6**	**2.8**
EER	Auxiliary (Liu et al. (2018c))	4.7	0.0	1.6	10.5	4.6	10.0	6.4	12.7	0.0	19.6	9.3	7.5	0.0	6.7
	SDTN (Liu et al. (2020))	2.1	2.2	0.0	7.2	0.1	3.9	4.8	**0.0**	0.0	19.6	5.3	5.4	**0.0**	4.8
	Step1	3.8	2.7	1.5	2.7	1.9	**1.8**	2.4	15.1	0.7	28.7	4.1	4.9	1.0	4.3
	Step1+Step2 w/ single trace	6.7	5.3	0.8	**1.5**	1.4	3.3	3.2	21.5	1.0	27.1	6.5	6.1	1.5	5.8
	Step1+Step2	2.4	3.1	0.4	2.6	1.2	3.0	2.4	9.5	0.4	23.5	1.1	0.5	0.6	2.8
	Step1+Step2+Step3 (Ours)	**2.5**	**1.0**	**0.0**	2.1	**1.0**	1.9	**2.2**	8.2	**0.0**	**18.5**	**0.8**	**0.0**	0.4	**2.5**
TPR@ FNR=.5%	SDTN (Liu et al. (2020))	**90.1**	76.1	80.7	71.5	62.3	74.4	85.0	**100**	100	33.8	49.6	30.6	97.7	70.4
	Step1	43.8	43.3	47.2	44.5	62.9	54.8	55.4	16.7	90.6	31.5	60.3	56.7	77.1	59.3
	Step1+Step2 w/ single trace	58.9	76.8	97.6	**94.2**	94.9	66.3	78.3	13.3	94.1	49.1	62.4	58.5	92.1	74.8
	Step1+Step2	84.7	74.7	100	70.1	**96.6**	77.5	89.6	36.9	100	40.1	96.3	99.4	99.4	89.7
	Step1+Step2+Step3 (Ours)	85.7	**85.4**	**100**	76.6	96.3	**80.2**	**93.8**	41.1	**100**	**55.8**	**98.1**	**100**	**99.8**	**91.2**

Table 4.3 The evaluation and ablation study on SiW-M Protocol I: known spoof detection.

Metrics (%)	Method	Replay	Print	3D Mask					Makeup			Partial Attacks			Average
				Half	Silic.	Trans.	Paper	Mann.	Ob.	Im.	Cos.	Fun.	Papergls.	Paper	
APCER	Auxiliary (Liu et al. (2018c))	23.7	7.3	27.7	18.2	97.8	8.3	16.2	100.0	18.0	16.3	91.8	72.2	0.4	38.3 ± 37.4
	LBP+SVM (Boulkenafet et al. (2017b))	19.1	15.4	40.8	20.3	70.3	0.0	4.6	96.9	35.3	**11.3**	53.3	58.5	0.6	32.8 ± 29.8
	DTL (Liu et al. (2019a))	**1.0**	0.0	0.7	24.5	58.6	0.5	3.8	73.2	13.2	12.4	17.0	17.0	0.2	17.1 ± 23.3
	CDCN (Yu et al. (2020b))	8.2	6.9	8.3	7.4	20.5	5.9	5.0	43.5	1.6	14.0	24.5	18.3	1.2	12.7 ± 11.7
	SDTN (Liu et al. (2020))	1.6	**0.0**	**0.5**	**7.2**	9.7	0.5	**0.0**	96.1	0.0	21.8	**14.4**	**6.5**	0.0	12.2 ± 26.1
	CDCN++ (Yu et al. (2020b))	9.2	6.0	4.2	7.4	18.2	**0.0**	5.0	39.1	0.0	14.0	23.3	14.3	0.0	10.8 ± 11.2
	HMP (Yu et al. (2020a))	12.4	5.2	8.3	9.7	13.6	**0.0**	2.5	**30.4**	0.0	12.0	22.6	15.9	1.2	$\mathbf{10.3 \pm 9.1}$
	Ours	10.0	4.9	5.3	16.7	**3.5**	2.0	2.8	92.8	**0.0**	37.5	33.7	23.2	0.2	17.9 ± 25.8
BPCER	LBP+SVM (Boulkenafet et al. (2017b))	22.1	21.5	21.9	21.4	20.7	23.1	22.9	21.7	12.5	22.2	18.4	20.0	22.9	21.0 ± 2.9
	DTL (Liu et al. (2019a))	18.6	11.9	29.3	12.8	13.4	8.5	23.0	11.5	9.6	16.0	21.5	22.6	16.8	16.6 ± 6.2
	SDTN (Liu et al. (2020))	14.0	14.6	13.6	18.6	18.1	8.1	13.4	10.3	9.2	17.2	27.0	35.5	11.2	16.2 ± 7.6
	CDCN (Yu et al. (2020b))	9.3	8.5	13.9	10.9	21.0	**3.1**	7.0	45.0	2.3	16.2	26.4	20.9	5.4	14.6 ± 11.7
	CDCN++ (Yu et al. (2020b))	12.4	8.5	14.0	13.2	19.4	7.0	6.0	45.0	1.6	14.0	24.8	20.9	3.9	14.6 ± 11.4
	HMP (Yu et al. (2020a))	13.2	6.2	13.1	10.8	16.3	3.9	**2.3**	34.1	**1.6**	13.9	23.2	17.1	2.3	12.2 ± 9.4
	Auxiliary (Liu et al. (2018c))	10.1	6.5	10.9	11.6	**6.2**	7.8	9.3	11.6	9.3	7.1	**6.2**	8.8	10.3	8.9 ± 2.0
	Ours	**3.8**	**6.3**	**4.4**	**5.5**	11.3	3.5	6.0	**6.6**	1.8	**2.7**	6.5	**8.0**	**1.1**	$\mathbf{5.7 \pm 2.8}$
ACER	LBP+SVM (Boulkenafet et al. (2017b))	20.6	18.4	31.3	21.4	45.5	11.6	13.8	59.3	23.9	16.7	35.9	39.2	11.7	26.9 ± 14.5
	Auxiliary (Liu et al. (2018c))	16.8	6.9	19.3	14.9	52.1	8.0	12.8	55.8	13.7	**11.7**	49.0	40.5	5.3	23.6 ± 18.5
	DTL (Liu et al. (2019a))	9.8	6.0	15.0	18.7	36.0	4.5	13.4	48.1	11.4	14.2	19.3	19.8	8.5	16.8 ± 11.1
	CDCN (Yu et al. (2020b))	8.7	7.7	11.1	**9.1**	20.7	4.5	5.9	44.2	2.0	15.1	25.4	19.6	3.3	13.6 ± 11.7
	SDTN (Liu et al. (2020))	7.8	7.3	7.1	12.9	13.9	4.3	6.7	53.2	4.6	19.5	20.7	21.0	5.6	14.2 ± 13.2
	CDCN++ (Yu et al. (2020b))	10.8	7.3	9.1	10.3	18.8	3.5	5.6	42.1	0.8	14.0	24.0	17.6	1.9	12.7 ± 11.2
	HMP (Yu et al. (2020a))	12.8	5.7	10.7	10.3	14.9	**1.9**	**2.4**	**32.3**	**0.8**	**12.9**	22.9	16.5	1.7	$\mathbf{11.2 \pm 9.2}$
	Ours	**6.9**	**5.6**	**4.8**	11.1	**7.4**	2.7	4.4	49.7	0.9	20.1	**20.1**	**15.6**	**0.6**	11.5 ± 13.2
EER	LBP+SVM (Boulkenafet et al. (2017b))	20.8	18.6	36.3	21.4	37.2	7.5	14.1	51.2	19.8	16.1	34.4	33.0	7.9	24.5 ± 12.9
	Auxiliary (Liu et al. (2018c))	14.0	4.3	11.6	12.4	24.6	7.8	10.0	72.3	10.1	**9.4**	21.4	18.6	4.0	17.0 ± 17.7
	DTL (Liu et al. (2019a))	10.0	**2.1**	14.4	18.6	26.5	5.7	9.6	50.2	10.1	13.2	19.8	20.5	8.8	16.1 ± 12.2
	CDCN (Yu et al. (2020b))	8.2	7.8	8.3	**7.4**	20.5	5.9	5.0	47.8	1.6	14.0	24.5	18.3	1.1	13.1 ± 12.6
	SDTN (Liu et al. (2020))	7.6	3.8	8.4	13.8	14.5	5.3	4.4	35.4	0.0	19.3	21.0	20.8	1.6	12.0 ± 10.0
	CDCN++ (Yu et al. (2020b))	9.2	5.6	**4.2**	11.1	19.3	5.9	5.0	43.5	0.0	14.0	23.3	14.3	**0.0**	11.9 ± 11.8
	HMP (Yu et al. (2020a))	13.4	5.2	8.3	9.7	13.6	5.8	**2.5**	**33.8**	0.0	14.0	23.3	16.6	1.2	11.3 ± 9.5
	Ours	**5.2**	4.4	4.4	10.1	**8.6**	**2.6**	4.3	47.2	**0.0**	19.6	**18.6**	**12.4**	0.7	$\mathbf{10.6 \pm 12.6}$
TPR@ FNR=.5%	SDTNLiu et al. (2020)	45.0	40.5	45.7	36.7	11.7	40.9	74.0	0.0	67.5	16.0	13.4	9.4	62.8	35.7 ± 23.9
	Ours	**55.1**	**46.4**	**57.3**	**65.1**	**33.0**	**91.7**	**76.7**	**0.0**	**100.0**	**46.4**	**31.8**	**15.4**	**97.7**	$\mathbf{53.7 \pm 31.8}$

Table 4.4 The evaluation on SiW-M Protocol II: unknown spoof detection.

noises and our model recognizes them as unknown spoof traces, which leads to an increased false negative rate (*i.e.*, BPCER). How to separate sensor noises from spoof traces can be an important future research topic.

SiW SiW Liu et al. (2018c) is another recent high-quality database. It includes fewer capture

cameras but more spoof mediums and environment variations, such as pose, illumination, and expression. The comparison on three protocols is shown in Tab. 4.2. We outperform the previous works on the first two protocols and rank in the middle on protocol 3. Protocol 3 aims to test the performance of unknown spoof detection, where the model is trained on one spoof attack (print or replay) and tested on the other. As we can see from Fig.4.8-4.9, the traces of print and replay are significantly different, where the replay traces are more on the high-frequency part (*i.e.*, trace component **T**) and the print traces are more on the low-frequency part (*i.e.*, trace component **S**). These pattern divergence leads to the adaption gap of our method while training on one attack and testing on the other.

SiW-M SiW-M Liu et al. (2019a) contains a large diversity of spoof types, including print, replay, 3D mask, makeup, and partial attacks. This allows us to have a comprehensive evaluation of the proposed approach with different spoof attacks. To use SiW-M for known spoof detection, we randomly split the data of all types into train/test set with a ratio of 60% vs. 40%, and the results are shown in Tab. 4.3. Compared to the preliminary version Liu et al. (2020), our method outperforms on most spoof types as well as the overall EER performance by 47.9% relatively, which demonstrates the superiority of our anti-spoofing on known spoof attacks.

For experiments on SiW-M (protocol I, II, and III), we additionally report the TPR at FNR equal to 0.5%. While EER and ACER provide the theoretical evaluation, the users in real-world applications care more about the true spoof detection rate under a given live detection error rate, and hence TPR can better reflect how well the model can detect one or a few spoof attacks in practices. As shown in Tab. 4.3, we improve the overall TDR of our preliminary version Liu et al. (2020) by 29.5%.

Metrics (%)	Method	Replay	Print	3D Mask					Makeup			Partial Attacks			Overall
				Half	Silic.	Trans.	Paper	Mann.	Ob.	Im.	Cos.	Funny.	Papergls.	Paper	
ACER	Auxiliary (Liu et al. (2018c))	6.7	5.6	8.5	7.5	11.6	6.7	6.4	8.9	5.7	6.1	14.3	15.9	5.4	8.4 ± 3.4
	Ours	**4.7**	**3.5**	**3.4**	**3.3**	**6.4**	**2.6**	**3.8**	**7.0**	**2.3**	**3.2**	**10.7**	**7.3**	**3.2**	$\mathbf{4.7 \pm 2.4}$
EER	Auxiliary (Liu et al. (2018c))	6.4	5.6	7.7	6.5	10.3	6.1	6.1	8.4	5.1	6.3	15.3	13.1	5.7	7.9 ± 3.2
	Ours	**4.1**	**2.8**	**3.4**	**3.1**	**5.6**	**3.6**	**3.0**	**6.7**	**2.2**	**3.4**	**10.2**	**8.6**	**2.2**	$\mathbf{4.5 \pm 2.5}$
TPR@ FNR=.5%	Auxiliary (Liu et al. (2018c))	60.4	65.5	64.4	70.4	47.5	67.0	71.6	64.3	75.1	69.8	45.8	47.8	62.9	62.5 ± 9.7
	Ours	**87.4**	**78.7**	**81.0**	**84.5**	**69.0**	**86.3**	**84.7**	**85.0**	**91.0**	**89.3**	**66.6**	**64.4**	**91.1**	$\mathbf{81.6 \pm 9.2}$

Table 4.5 The evaluation on SiW-M Protocol III: openset spoof detection.

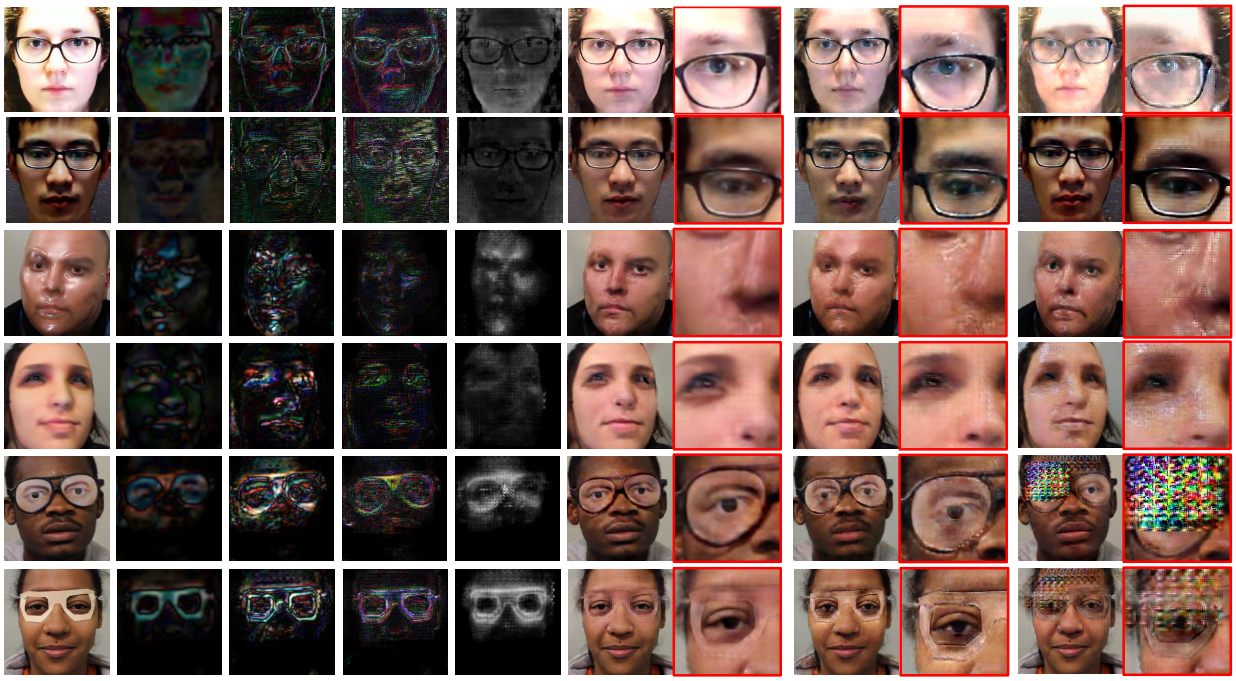


Figure 4.8 Examples of each spoof trace components. (a) the input sample faces. (b) **B**. (c) **C**. (d) **T**. (e) **P**. (f) the final live counterpart reconstruction and zoom-in details. (g) results from Liu et al. (2020). (h) results from Step1+Step2 with a single trace representation.

4.4.3 Anti-Spoofing for Unknown and Open-set Spoofs

Another important aspect is to test the anti-spoofing performance on unknown spoof. To use SiW-M for unknown spoof detection, Liu et al. (2019a) defines the leave-one-out testing protocols, termed as SiW-M Protocol II. In this protocol, each model (*i.e.*, one column in Tab. 4.4) is trained with 12 types of spoof attacks (as known attacks) plus the 80% of the live faces, and tested on the remaining 1 attack (as unknown attack) plus the 20% of live faces. As shown in Tab. 4.4, our PhySTD achieves significant improvement over our preliminary version, with relatively 11.7% on

Metrics(%)	Attacks	Protocol I	Protocol II	Protocol III
ACER	Impersonation	2.2	4.0	3.3
	Obfuscation	4.6	14.9	5.4
EER	Impersonation	1.4	3.1	3.2
	Obfuscation	3.6	14.0	5.1
TPR@ FNR=.5%	Impersonation	87.8	73.3	85.9
	Obfuscation	84.6	47.0	79.5

Table 4.6 The performance comparison between impersonation attacks and obfascation attacks.

the overall EER, 19.0% on the overall ACER, 50.4% on the overall TPR. Specifically, we reduce the EERs of half mask, paper glasses, transparent mask, replay attack, and partial paper relatively by 47.6%, 40.4%, 37.7%, 31.6%, 56.3%, respectively. Overall, compared with the top 7 performances, we outperform the SOTA performance of EER/TPR and achieve comparable ACER. Among all, the detection of silicone mask, paper-crafted mask, mannequin head, impersonation makeup, and partial paper attacks are relatively good, with the detection accuracy (*i.e.*, TPR@FNR=0.5%) above 65%. Obfuscation makeup is the most challenging one with TPR of 0%, where we predict all the spoof samples as live. This is due to the fact that the makeup looks very similar to the live faces, while being dissimilar to any other spoof types. However, once we obtain a few samples, our model can quickly recognize the spoof traces on the eyebrow and cheek, synthesize new spoof samples, and successfully detect the attack (TPR=41.1% in Tab. 4.3).

Moreover, in the real-world scenario, the testing samples can be either a known spoof attack or an unknown one. Thus, we propose SiW-M Protocol III to evaluate this open-set testing situation. In Protocol III, we first follow the train/test split from protocol I, and then further remove one spoof type as the unknown attack. During the testing, we test on the entire unknown spoof samples as well the test split set of the know spoof samples. The results are reported in Tab. 4.5. Compared to the SOTA face anti-spoofing method Liu et al. (2018c), our approach substantially outperforms it in all three metrics.

Impersonation *v.s.* Obfuscation In Tab. 4.6, we show the comparison of our performance towards

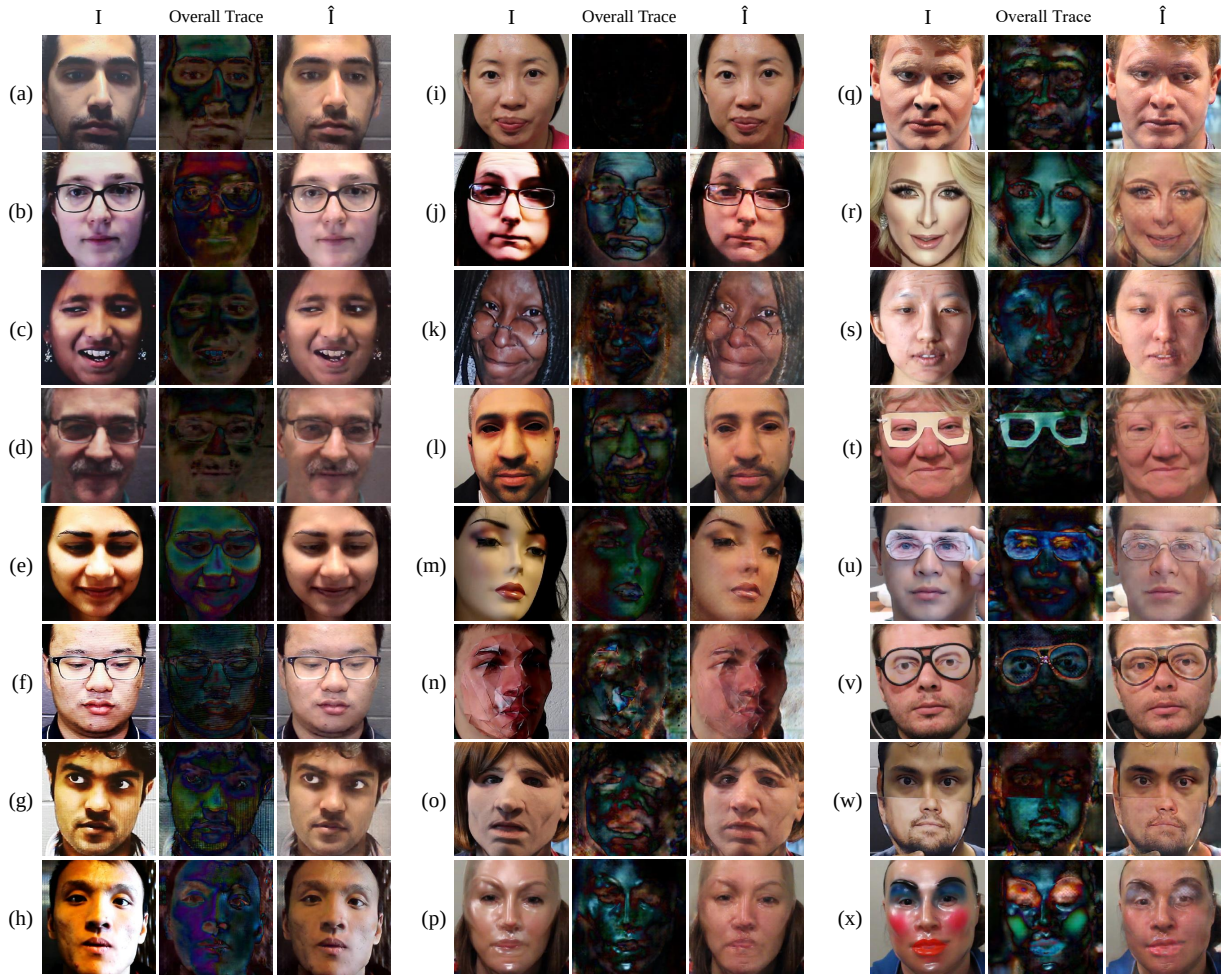


Figure 4.9 Examples of spoof trace disentanglement on SiW (a-h) and SiW-M (i-x). (a)-(d) items are print attacks and (e)-(h) items are replay attacks. (i)-(x) items are live, print, replay, half mask, silicone mask, paper mask, transparent mask, obfuscation makeup, impersonation makeup, cosmetic makeup, paper glasses, partial paper, funny eye glasses, and mannequin head. The first column is the input face, the second column is the overall spoof trace ($\mathbf{I} - \hat{\mathbf{I}}$), the third column is the reconstructed live.

impersonation attacks and obfuscation attacks on 3 protocols in SiW-M database. On all 3 protocols, we see a better performance on impersonation attacks over obfuscation attacks, especially in the Protocol II unknow attack situations. Obfuscation attacks generally have a larger appearance discrepancy compared to impersonation attacks, and it's naturally more difficult for the CNN model to do out-of-distribution predictions. In practice, most of the attacks are impersonation attacks, and hence our solution can be effective to the practical situations.

Label \ Predict	Live	Print1	Print2	Replay1	Replay2
Live	56(−4)	1(+1)	1(+1)	1(+1)	1(+1)
Print1	0	43(+2)	11(+9)	3(−8)	3(−3)
Print2	0	9(−25)	48(+37)	1(−8)	2(−4)
Replay1	1(−9)	2(−1)	3(+3)	51(+38)	3(−28)
Replay2	1(−7)	2(−5)	2(+2)	3(−3)	52(+13)

Table 4.7 Confusion matrices of spoof mediums classification based on spoof traces. The results are compared with the previous method Jourabloo et al. (2018). Green represents improvement over Jourabloo et al. (2018). Red represents performance drop.

Label \ Predict	Live	Print	Replay	Masks	Makeup	Partial
Live	116	6	6	3	0	0
Print	1	40	1	3	0	1
Replay	3	1	32	1	0	1
Masks	3	1	1	90	0	3
Makeup	3	0	0	0	36	0
Partial	2	0	0	2	0	146

Table 4.8 Confusion matrices of 6-class spoof traces classification on SiW-M database.

4.4.4 Spoof Traces Classification

To quantitatively evaluate the spoof trace disentanglement, we perform a spoof medium classification on the disentangled spoof traces and report the classification accuracy. The spoof traces should contain spoof medium-specific information, so that they can be used for clustering without seeing the face. To make a fair comparison with Jourabloo et al. (2018), we remove the additional spoof type information from the preliminary mask $\mathbf{P}_0$. That is, for this specific experiment, we only use the additive traces $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ to learn the trace classification. After $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ finish training with only binary labels, we fix PhySTD and apply a simple CNN (*i.e.*, AlexNet) on the estimated additive traces to do a supervised spoof medium classification. We follow the same 5-class testing protocol in Jourabloo et al. (2018) in Oulu-NPU Protocol 1. We report the classification accuracy as the ratio between correctly predicted samples from all classes and all testing samples. Shown in Tab. 4.7. Our model can achieve a 5-class classification accuracy of 83.3%. If we treat two print attacks as the

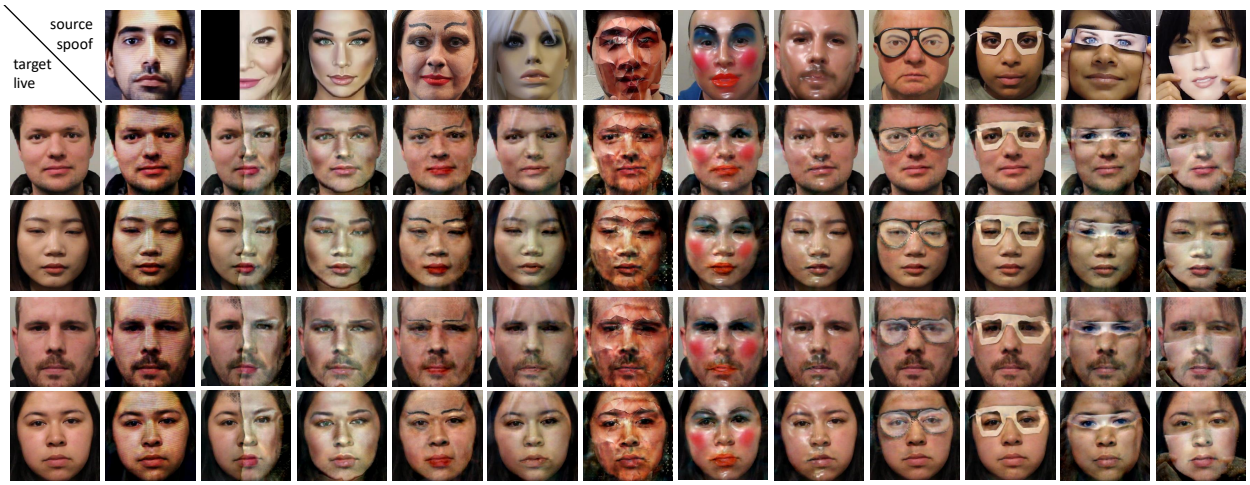


Figure 4.10 Examples of the spoof data synthesis. The first row are the source spoof faces, the first column are the target live faces, and the remaining are the synthesized spoof faces from the live face with the corresponding spoof traces.

same class and two replay as the same class, our model can achieve a 3-class classification accuracy of 92.0%. Compared with the prior method Jourabloo et al. (2018), we show an improvement of 29% on the 5-class model. In addition, we train the same CNN on the original images instead of the estimated spoof traces for the same spoof medium classification task, and the classification accuracy can only reach 80.6%. This further demonstrates that the estimated traces do contain significant information to distinguish different spoof mediums.

We also execute the spoof traces classification task on more spoof types in SiW-M database. We leverage the train/test split on SiW-M Protocol 1. We first train the PhySTD till convergence, and use the estimated traces from the training set to train the trace classification network. We explore the 6-class scenario, shown in Tab. 4.8. Our 6-class model can achieve the classification accuracy of 92.0%. Since the traces are more distinct among different spoof types, this performance is even better than 5-class classification on print/replay scenario in Oulu-NPU Protocol 1. This further demonstrates that PhySTD can estimate spoof traces that contain significant information of spoof mediums and can be applied to multiple spoof types.

4.4.5 Ablation Study

In this section, we show the importance of each design of our proposed approach on the SiW-M Protocol I, in Tab.4.3. Our baseline is the auxiliary FAS Liu et al. (2018c), without the temporal module. It consists of the backbone encoder and depth estimation network. When including the image decomposition, the baseline becomes the training step 1 in Alg. 1, as the traces are not activated without the training step 2. To validate the effectiveness of GAN training, we report the results from the baseline model with our GAN design, denoted as Step1+Step2. We also provide the control experiment where the traces are represented by a single component to demonstrate the effectiveness of the proposed 5-element trace representation. This model is denoted as Step1+Step2 with single trace. In addition, we evaluate the effect of training with more synthesized data via enabling the training step 3 as Step1+Step2+Step3, which is our final approach.

As shown in Tab. 4.3, the baseline model (Auxiliary) can achieve a decent performance of EER 6.7%. Adding image decomposition to the baseline (Step 1) can improve the EER from 6.7% to 4.3%, but more live samples are predicted with higher scores, causing a worse ACER. Adding simple GAN design (Step1+Step2 with single trace) may lead to a similar EER performance of 5.8%, but based on the TPR ($59.3\% \rightarrow 74.8\%$) its practical performance may be improved. With the proper physics-guided trace disentanglement, we can improve the EER to 2.8% and TPR to 89.7%. And our final design can achieve the performance of HTER 2.8%, EER 2.5%, and TPR 91.2%. Compared with our preliminary version, the EER is improved by 47.9%, HTER is improved by 31.7% and TPR is improved by 29.5%.

4.4.6 Visualization

Spoof trace components In Fig.4.8, we provide illustration of each spoof trace component. Strong

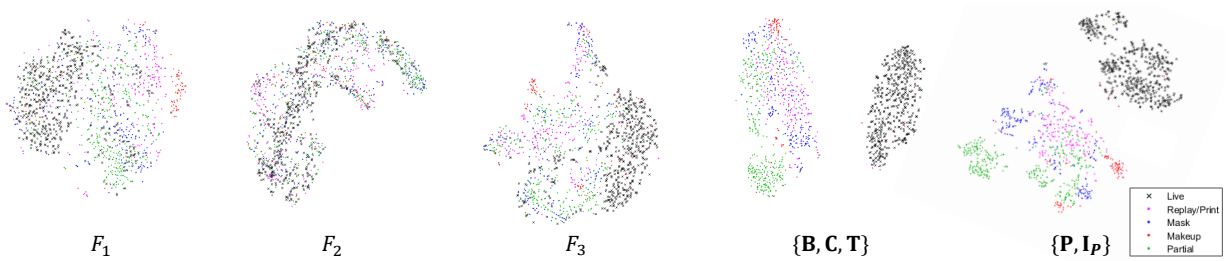


Figure 4.11 The tSNE visualization of features from different scales and layers. The first 3 visualization are from the encoder feature F_1,F_2,F_3, and the last 2 visualization are from the features that produce $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ and $\{\mathbf{P}, \mathbf{I}_P\}$.

color distortion (low-frequency trace) shows up in the print attacks. Moir patterns in the replay attack are well detected in the high-frequency trace. The local specular highlights in transparent mask are well presented in the low- and mid-frequency components, and the inpainting process further fine-tunes the most highlighted area. For the two glasses attacks, the color discrepancy is corrected in the low-frequency trace, and the sharp edges are corrected in the mid- and high-frequency traces. Each component shows a consistent semantic meaning on different spoof samples, and this successful trace disentanglement can lead to better final visual results. As shown on the right side of Fig. 4.8, we compare with our preliminary version Liu et al. (2020) and the ablated GAN design with a single trace representation. The result of single trace representation shows strong artifacts on most of the live reconstruction. The multi-scale from our preliminary version has already shown a large visual quality improvement, but still have some spoof traces (*e.g.*, glass edges) remained in the live reconstruction. In contrast, our approach can further handle the missing traces and achieve better visualization.

Live reconstruction In Fig. 4.9, we show more examples from different spoof types in SiW and SiW-M databases. The overall trace is the exact difference between the input face and its live reconstruction. For the live faces, the trace is zero, and for the spoof faces, our method removes spoof traces without unnecessary changes, such as identity shift, and make them look like live faces. For example, strong color distortion shows up in print/replay attacks (Fig. 4.9a-h) and some 3D

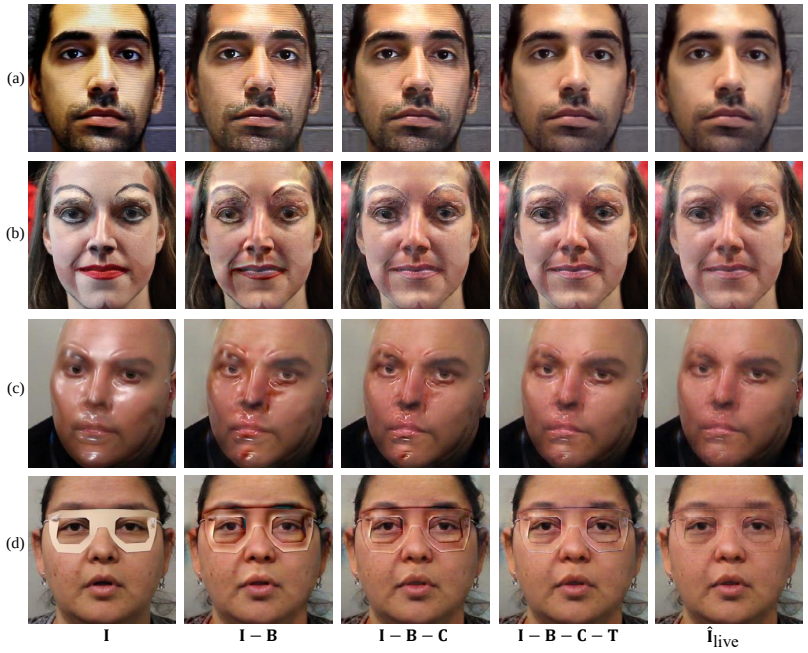


Figure 4.12 The illustration of removing the disentangled spoof trace components one by one. The estimated spoof trace elements of input spoof (the first column) are progressively removed in the order of $\mathbf{B}, \mathbf{C}, \mathbf{T}, \mathbf{T_P}$. The last column shows the reconstructed live image after removing all three additive trace components and the inpainting trace. (a) Replay attack; (b) Makeup attack; (c) Mask attack; (d) Paper glasses attack.

mask attacks (Fig. 4.9l-o). For makeup attacks (Fig. 4.9q-s), the fake eyebrows, lipstick, artificial wax, and cheek shade are clearly detected. The folds and edges (Fig. 4.9t-w) are well detected and removed in paper-crafted masks, paper glasses, and partial paper attacks.

Spoof synthesis Additionally, we show examples of new spoof synthesis using the disentangled spoof traces, which is an important contribution of this work. As shown in Fig. 4.10, the spoof traces can be precisely transferred to a new face without changing the identity of the target face. Due to the additional inpainting process, spoof attacks such as transparent mask and partial attacks can be better attached to the new live face. Thanks to the proposed 3D warping layer, the geometric discrepancy between the source spoof trace and the target face can be corrected during the synthesis. Especially on the second source spoof, the right part of the traces is successfully transferred to the new live face while the left side remains to be still live. It demonstrates that our trace regularization can suppress unnecessary artifacts generated by the network. Both the live reconstruction results

Figure 4.13 The illustration of double spoof trace disentangling. The left 4 samples are live faces, and the right 4 samples are spoof faces. (a) Original Input. (b) 1st round live reconstruction. (c) 1st round spoof traces. (d) 2nd round live reconstruction. (e) 2nd round spoof traces.

in Fig. 4.9 and the spoof synthesis results in Fig. 4.10 demonstrate that our approach disentangles visually convincing spoof traces that help face anti-spoofing.

Spoof trace removing process As shown in Fig. 4.12, we illustrate the effects of trace components by progressively removing them one by one. For the replay attack, the spoof sample comes with strong over-exposure as well as clear Moir pattern. Removing the low-frequency trace can effectively correct the over-exposure and color distortion caused by the digital screen. And removing the texture pattern in the high-frequency trace can peel off the high-frequency grid effect and reconstruct the live counterpart.

For the makeup attack, since there is no strong color range bias, removing estimated low-frequency trace would mainly remove the lip-stick color and fake eyebrow, but in the meantime bring a few artifacts at the edges. Next, while removing the content pattern, the shadow on the cheek and the fake eyebrows are adequately lightened. Finally, removing the texture pattern would significantly correct the spoof traces from artificial wax, eyeliner, and shadow on the cheek. Similarly, in mask and partial attacks, the reconstruction will be gradually refined as we removing

components one by one.

To validate the quality of spoof trace removing, we execute a double spoof trace disentanglement, shown in Fig. 4.13. For each sample, we execute a 2-round disentanglement, where the second-round input is the live reconstruction from the first round. As we can see from the figure, regardless of the original spoofness, the networks recognize all live reconstruction as live, which shows high confidence on the first round spoof trace removal. The 1st-round average spoof score for live faces is 0.11, and the second round is 0.03. The 1st-round average spoof score for spoof faces is 0.89, and the second round is 0.03. However, we can still see different degrees of spoof traces left in the reconstructed live faces. How to effectively measure the quality of spoof trace removing and use it for a second-round supervision can be a future research.

To validate the relation of spoof trace intensity and the spoofness score, we execute a reverse double spoof trace disentanglement, where the input of the second round disentanglement is the original input with only 50% of the estimated spoof traces removed. For this experiment, the second round score for live faces changed from 0.03 t0 0.07, and the second round score for spoof faces changed from 0.03 to 0.53. Based on the results, we can tell that the spoof trace intensity is positively correlated with the spoofness score.

t-SNE visualization We use t-SNE Maaten & Hinton (2008) to visualize the encoder features $\mathbf{F}_1$,$\mathbf{F}_2$,$\mathbf{F}_3$, and the features that produce $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ and $\{\mathbf{P}, \mathbf{I}_P\}$. The t-SNE is able to project the output of features from different scales and layers to 2D by preserving the KL divergence distance. As shown in Fig. 4.11, among the three feature scales in the encoder, F_3 is the most separable feature space, the next is F_1, and the worst is F_2. The features for additive traces $\{\mathbf{B}, \mathbf{C}, \mathbf{T}\}$ are well-clustered as semantic sub-groups of live, makeup, mask, and partial attacks. As we know the inpainting masks for live samples are close to zero, the feature for inpainting traces $\{\mathbf{P}, \mathbf{I}_P\}$ shows the inpainting process mostly update the partial attacks, and then some makeup attacks and

mask attacks, *i.e.*, the green dots being further away from the black dots means they have greater magnitude. This validates our prior knowledge of the inpainting process.

4.5 Conclusions

This work proposes a physics-guided spoof traces disentanglement network (PhySTD) to tackle the challenging problem of disentangling spoof traces from the input faces. With the spoof traces, we reconstruct the live faces as well as synthesize new spoofs. To correct the geometric discrepancy in synthesis, we propose a 3D warping layer to deform the traces. The disentanglement not only improves the SOTA of face anti-spoofing in known, unknown, and open-set spoof settings, but also provides visual evidence to support the model's decision. To note that, even though the proposed spoof trace modeling is based on a physical approximation of the spoof presentation attack, the whole learning process is still intensely relying on the data. In addition, our visualization/interpretation of spoof attacks is on the image appearance, rather than the feature level, such as Grad-CAM.

Chapter 5

Visualization: Blind Removal of Facial Foreign Shadow

5.1 Introduction

In our daily activities, many external objects around us can cast shadows on faces, termed as *facial foreign shadow*. For instance, while we take a selfie outdoors, our hand and camera might block part of the sunlight and create a shadow on the face. Dynamic and scattered shadows may be produced by leaves while walking under trees. During driving, the driver may confront the high-contrast lighting caused by the direct sunlight and car pillars. While people may experience these situations everyday, the shadow cast sometimes can be unwanted. In some cases, the shadows cast should be removed for aesthetic purposes, such as photoshop and face editing. In others, the shadows cast could negatively impact face-related tasks, such as face recognition, expression analysis, age estimation, and driver monitoring.

While facial foreign shadow removal is a relatively new topic, there are a few related studies. Many works aim to handle the self shadow and relight the face under a different lighting, via quotient image Wen et al. (2003); Zhou et al. (2019), inverse rendering Nagano et al. (2019); Nestmeyer et al. (2020), and style transfer Gu et al. (2019); Lee et al. (2020). Those methods focus more on the global lighting distribution and might be limited in handling arbitrary high-frequency structure caused by harsh foreign shadows. There are also face completion works under structured

Figure 5.1 The results of our shadow removal model on images from our Shadow Face in the Wild (SFW) database (*top*) and UCB database Zhang et al. (2020b) (*bottom*). The left to right are input face, output face, and shadow matte.

occlusions, such as square, circle, and lattice Li et al. (2020); Yang et al. (2019b); Zhang et al. (2017b). Compared with foreign shadow removal, face completion is relatively easier as the shape is less complicated and the occlusion is often filled with a single color such as white. Further, some works study shadow on generic objects Le & Samaras (2019); Shor & Lischinski (2008). While they excel at shadow detection, when applied to faces, observable artifacts can be detected on de-shadow results due to the lack of face priors.

The major problem of these prior methods is that they cannot handle the high-frequency structure caused by the harsh shadows as demonstrated in Sun et al. (2019); Zhang et al. (2020b). Instead of predicting illumination, Xuaner *et al.* propose a single image-based approach using only perceptual and pixel intensity losses and train the network on a synthetic shadow dataset Zhang et al. (2020b). It turns out the pixel intensity loss works better to remove harsh shadows and recover fine details. However, the model based only on perceptual and pixel intensity losses does not generalize well in practice, as it is hard to build the training dataset covering real-world complex lighting conditions.

As stated above, this chapter aims to detect and remove the foreign shadow from in-the-wild

faces. While we focus on the foreign shadow, we also like to address strong self-shadow caused by self occlusion (see Fig. 5.2). To tackle this problem, we face three major challenges. First, the shadow of in-the-wild faces is arbitrary, varying from different sizes and shapes, to different locations, colors, blurriness and intensities. Prior works Le & Samaras (2019); Shor & Lischinski (2008); Zhang et al. (2020b) model the shadow directly in the RGB space. Given the level of diversity, they have a hard time to address all the discrepancies, leaving some observable artifacts in de-shadowed faces. Second, there are few public databases for training and evaluation. To capture the paired shadow and non-shadow faces, both the subject and photographer need to be perfectly still, which is rarely feasible. Third, sometimes the shadow removal is extended from single images to video. On one hand, multiple frames (*e.g.*, live photo) may provide addition cues to benefit single-image shadow removal. On the other hand, video shadow removal requires additional temporal consistency.

To address the aforementioned challenges, we propose a novel blind removal model of facial foreign shadow. To handle the shadow diversity, we propose a simple yet effective approach to decompose the direct RGB shadow removal into grayscale shadow removal and colorization. We show that, without color, the shadow modeling becomes a much simpler task and the grayscale removal model is easy to generalize to unseen data. After that, with the knowledge of shadow region from grayscale shadow removal, the colorization is turned into an image inpainting process. Without seen the biased color information from shadow region, the colorization process also becomes more generalizable. Moreover, to ensure the temporal consistency, we propose a temporal sharing module (TSM) to aggregate the information among multiple frames. TSM includes an efficient warping layer, in order to handle frames with pose and expression variations.

For training the model, we follow the process proposed by Zhang et al. (2020b) to built a synthetic database that contains paired shadow and shadow-free faces. Foreign shadows are generated

(a) (b) (c)

Figure 5.2 Examples of (a) foreign shadow, (b) strong self shadow, and (c) normal self shadow. Our model is designed to remove unwanted shadows in (a-b) while keeping normal shadow in (c).

with randomized properties and moving trajectories. Further, we collect a face database with 280 videos captured under highly dynamic environments for evaluation purposes. The external objects casting shadows include hands, books, leaves, trees, window blinds, car pillars, and buildings. To quantitative evaluate shadow segmentation, we provide detailed pixel-level segmentation annotation for this database.

In summary, the main contributions of this chapter include:

◇ A novel approach to decompose RGB shadow removal into grayscale shadow removal and colorization;

◇ A temporal sharing module to ensure video consistency;

◇ A face shadow database under dynamic environments;

◇ SOTA results and photo-realistic de-shadow quality.

5.2 Related Work

Face relighting Face relighting methods could be roughly divided into three categories, quotient image-based, style transfer, and inverse rendering. The color ratio (*i.e.*, quotient images) is first proposed in Shashua & Riklin-Raviv (2001) to transfer a frontal face from one lighting to another. This basic idea has been extended to handle different poses, use ratios of radiance environment maps, and generate synthetic relighting dataset in Stoschek (2000); Wen et al. (2003); Zhou et al. (2019). Facial lighting also can be changed by style transfer Gu et al. (2019); Lee et al. (2020); Liao et al. (2017); Shih et al. (2014); Shu et al. (2017a); Sun et al. (2019). Similar to quotient images, style transfer methods need at least a reference image as the target style. Moreover, the face poses of input and reference images are often very close. In the category of inverse rendering, a face image could be decomposed into multiple components such as geometry, reflectance, and lighting Egger et al. (2018); Nagano et al. (2019); Nestmeyer et al. (2020); Sengupta et al. (2018); Shu et al. (2017b); Tewari et al. (2017); Tran & Liu (2018); Wang et al. (2008). For example, in Nestmeyer et al. (2020), the intrinsic components (*e.g.* normal and albedo) are predicted and combined through diffuse rendering in the first network, the second network is then applied to learn non-diffuse residual. In general, inverse rendering methods rely on multiple sub-networks for the decomposition, which are not sufficiently efficient to handle facial foreign shadows in videos.

Face Completion Face completion aims to fill in the missing or occluded face regions with semantic meaningful information. In Zhang et al. (2017b), Zhang *et al.* proposed a DemeshNet with two sub-networks to remove mesh-like lines or watermarks on faces. Li *et al.* proposed a disentangling and fusing network that contains discriminators in three domains, *i.e.*, occluded faces, clean faces, and structured occlusions Li et al. (2020). The face inpainting network in Yang et al. (2019b) comprises of a landmark predicting subnet and an inpainting subnet. Different from the shadow

removal, the structured occlusions either are opaque or contain repeated patterns. The networks are mainly used to hallucinate the invisible face regions.

Generic Shadow Detection and Removal Without much training data, early works in general-purpose shadow detection and removal mainly study shadow properties, especially around shadow edges Chuang et al. (2003); Finlayson et al. (2002); Wu et al. (2012); Wu & Tang (2005). For example,Wu et al. (2012) applied the graph-cut inference to detect shadow regions, and then used the shadow matting to generate soft shadow boundaries. Deep learning based methods have been proposed recently to detect and remove shadows Ding et al. (2019); Le & Samaras (2019); Qu et al. (2017); Wang et al. (2018a). Hu et al. (2018) designed the direction-aware spatial context module and applied a spatial RNN to detect shadows. Cun et al. (2020) learned to hierarchically aggregates the dilated multi-contexts and attentions. Authors in Zhang et al. (2020b) demonstrated that the general-purpose methods such as Cun et al. (2020); Hu et al. (2018) cannot preserve the authenticity of the input faces. One reason is that these general-purpose networks are unable to capture the specific face characteristics. For instance, human face skin is a highly scattering material that also has a complex absorption spectrum Donner & Jensen (2006). In this work, we propose a novel two-stage shadow modeling that can better handle both subsurface scattering effects and color distortion.

5.3 Proposed Method

5.3.1 Shadow synthesis and modeling

Shadow is produced by a foreign object that blocks part of the light rays from arriving to the face. Let a matte M represent the shadow shape, the shadow formation can be modeled as a blending

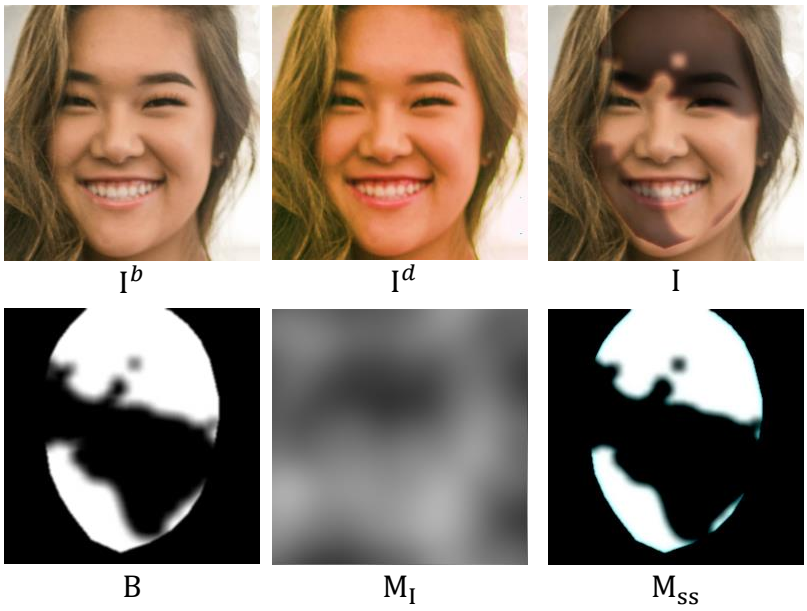


Figure 5.3 Illustration of data synthesis components.

between the well-illuminated face $\mathbf{I}^b$ and the under-illuminated face $\mathbf{I}^d$:

$$\mathbf{I} = \mathbf{I}^b \odot (1-\mathbf{M}) + \mathbf{I}^d \odot \mathbf{M}, \tag{5.1}$$

where $\odot$ denotes element-wise multiplication. As the real-world shadow varies in both shapes and intensities, it's vital to have $\{\mathbf{I}, \mathbf{I}^b\}$ paired data with a large variety of $\mathbf{M}$ to train a generalized shadow removal model. However, it's hardly feasible to collect a large-scale dataset with such pairing, as the subject needs to be perfectly static while capturing the pair. Therefore, creating a synthetic dataset becomes our go-to approach to tackle this problem.

Shadow synthesis As indicated in Zhang et al. (2020b), using Eqn. 5.1 to synthesize natural face shadow shall include additional variations: shape, intensity, subsurface scattering and color. Let shape $\mathbf{B}$ be a binary mask that defines the whole region affected by foreign shadow. The shadow is often unevenly distributed, such as in mottled patterns or gradually changed patterns, depending on the relative distance between the object and face, and the environmental lighting. We use a

gray-scale matte $\mathbf{M}_{\mathrm{I}}$ to represent the uneven intensity. In addition, the light outside the shadow region would penetrate beneath the skin, reach the vessels and reflect back, creating a red band around the shadow boundary. We represent such subsurface scattering effect by $\mathbf{M}_{\mathrm{ss}}$, which is computed by blurring $\mathbf{B}$ with different kernel per RGB channel. Therefore, Eqn. 5.1 can be updated to:

$$\mathbf{I} = \mathbf{I}^b \odot (1 - \mathbf{B} \odot \mathbf{M}_{\mathrm{ss}}) + \mathbf{I}^d \odot \mathbf{B} \odot \mathbf{M}_{\mathrm{ss}} \odot \mathbf{M}_{\mathrm{I}}. \tag{5.2}$$

Moreover, the shadow region may be under certain color distortion, due to the block of parts of light. We formulate such color distortion by a 3×3 color transfer matrix $\mathbf{C}$:

$$\mathbf{I}^d = \mathbf{I}^b \mathbf{C}. \tag{5.3}$$

The $\mathbf{B}$, $\mathbf{M}_{\mathrm{I}}$, $\mathbf{M}_{\mathrm{ss}}$, and $\mathbf{C}$ are illustrated in Fig. 5.3. During the synthesis, given a well-illuminated face $\mathbf{I}^b$, we generate random parameters for each component to synthesize different shadow faces $\mathbf{I}$, which is detailed in Sec. 5.4.

Shadow modeling With synthetic pairwise data, we can train a model $G(\cdot)$ to detect and remove foreign shadows ($\mathbf{I} \rightarrow \mathbf{I}^b$). Despite the complexity of shadow synthesis process, prior works Le & Samaras (2019); Shor & Lischinski (2008); Zhang et al. (2020b) opt to simplify the relation between $\mathbf{I}$ and $\mathbf{I}^b$ in $G(\cdot)$ as:

$$\mathbf{W}, \mathbf{N} \leftarrow G(\mathbf{I} \mid \boldsymbol{\omega}), \tag{5.4}$$

$$\hat{\mathbf{I}}^b = \mathbf{I} \odot \mathbf{W} + \mathbf{N}, \tag{5.5}$$

where $\boldsymbol{\omega}$ are the parameters of the shadow removal model, and both the scaling $\mathbf{W}$ and offset $\mathbf{N}$ are of the same size as $\mathbf{I}$. The motivation of this simplification is two-fold: 1) precisely estimating all shadow components (*i.e.*, $\mathbf{B}$, $\mathbf{M}_{\mathrm{I}}$, $\mathbf{M}_{\mathrm{ss}}$, $\mathbf{C}$) can be very challenging, and 2) even with full supervision

of all the components, reversing shadow formation may raise a convergence issue. This is due to the ambiguity in the shadow parameterization, where one shadow can be generated from different combinations of shadow components.

However, the prior works based on Eqn. 5.5 have a hard time to address all the discrepancies between the shadow and non-shadow regions, leaving some observable artifacts in the de-shadowed faces. We observe that it is not straightforward to derive from Eqn. 5.1 to Eqn. 5.5. Due to the existing of color transfer matrix $\mathbf{C}$, $\mathbf{W}$ and $\mathbf{N}$ themselves become a function of $\mathbf{I}^b$, instead of independent to $\mathbf{I}^b$. Thus, the model learning becomes a *chicken-and-egg* problem, which may easily turn into a memorization mode (*i.e.*, a type of learning that generalizes poorly Corneanu et al. (2019)). To tackle this issue, we propose to decompose the color shadow removal into grayscale shadow removal and colorization. While dealing with shadow removal in grayscale, $\mathbf{C}$ in Eqn. 5.3 simply becomes a scalar, and hence both $\mathbf{C}$ and $\mathbf{M}_{\text{ss}}$ can be integrated into $\mathbf{M}_{\text{I}}$ as $\mathbf{M}'_{\text{I}}$. We can then transfer the relation of Eqn. 5.1 into:

$$\begin{aligned}\hat{\mathbf{I}}^{b,gs} &= \mathbf{I}^{gs} \odot (1 - \mathbf{B}) + \mathbf{I}^{gs} \odot \mathbf{B} \oslash \mathbf{M}'_{\text{I}} \\ &= \mathbf{I}^{gs} \odot (1 - \mathbf{B} + \mathbf{B} \oslash \mathbf{M}'_{\text{I}}) \\ &= \mathbf{I}^{gs} \odot \mathbf{W},\end{aligned} \tag{5.6}$$

where $\hat{\mathbf{I}}^{b,gs}$ and $\mathbf{I}^{gs}$ are the grayscale version of $\mathbf{I}^b$ and $\mathbf{I}$, $\oslash$ is element-wise division, and $\mathbf{W} = 1 - \mathbf{B} + \mathbf{B} \oslash \mathrm{M}'_{\text{I}}$. It's clear that Eqn. 5.6 is in a close form and well aligned with Eqn. 5.5. As $\mathbf{W}$ and $\mathbf{N}$ are detached with $\mathbf{I}^b$, they are easier to learn. Next, we simply need to colorize the grayscale face to get the final face recovery in RGB. With the knowledge provided by the grayscale shadow removal, we turn the blind color recovery into a mask-guided image inpainting.

The overall pipeline is shown in Fig. 5.4. Our approach consists of three major steps: 1*)*

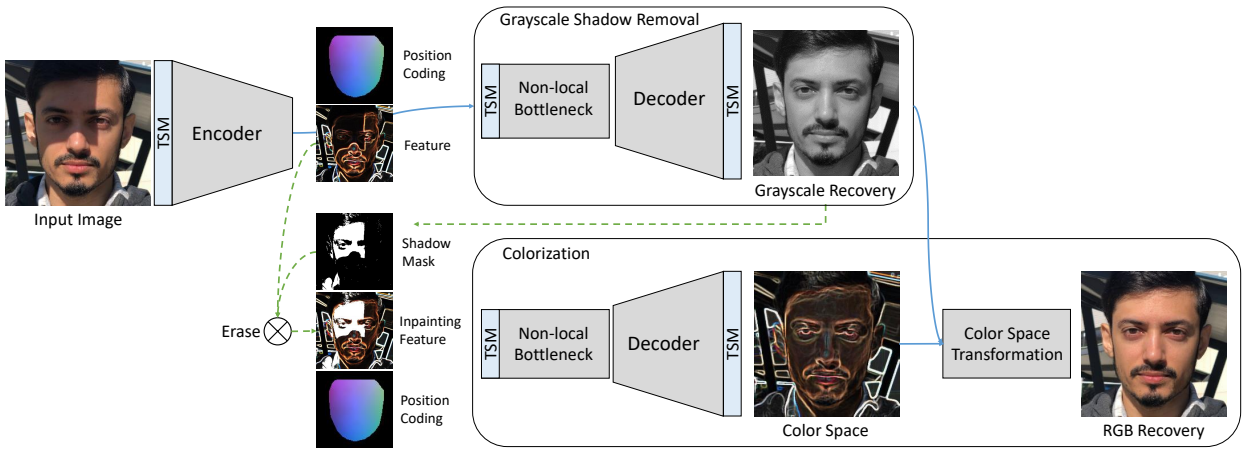


Figure 5.4 Illustration of our network architecture. The model mainly consists of an encoder, a shadow matte decoder, a color matrix decoder, and a shadow residual decoder. The Temporal Sharing Module (TSM) can be easily plugged into the face encoder. Together with the temporal consistency loss $\mathcal{L}_T$, we can leverage the unlabeled image frames efficiently. The green dashed lines indicate the short-cut connections and the orange dashed lines and boxes indicate the loss functions.

grayscale shadow removal (Sec. 5.3.2), 2*) colorization* (Sec. 5.3.3), and 3*) temporal information sharing* (Sec. 5.3.4). Step 1 and 2 are the key ingredients for a single frame shadow removal, and step 3 is the key ingredient for a smooth video shadow removal. In Sec. 5.3.5, we discuss the losses and training strategies in detail.

5.3.2 Grayscale shadow removal

The grayscale shadow removal module takes a RGB face $\mathbf{I} \in \mathbb{R}^{N^2 \times 3}$ as input, and outputs the scaling map $\mathbf{W} \in \mathbb{R}^{N^2 \times 1}$ and offset map $\mathbf{N} \in \mathbb{R}^{N^2 \times 1}$ that can recover a well-illuminated grayscale face $\hat{\mathbf{I}}^{b,gs} \in \mathbb{R}^{N^2 \times 1}$ based on Eqn. 5.5. The module consists of an encoder, a stack of residual non-local blocks, and a decoder. The encoder extracts features as $\mathbf{F}$ from input images for shadow removal. It contains 4 convolution layers and 3 times of downsampling. To encourage a spatial consistency of facial lighting and albedo, we leverage the latest design of non-local block and visual transformer Carion et al. (2020); Dosovitskiy et al. (2020); Wang et al. (2018c). We stack 3 residual non-local blocks to process the encoder features with position encoding. After that, the decoder

upsamples the features from non-local blocks via 3 transpose convolution layers, and estimate $\mathbf{W}$ and $\mathbf{N}$. We adopt a short-cut connection at each feature scale to bypass high-frequency information.

For the position encoding, we adopt the projected normalized coordinate code (PNCC) Zhu et al. (2017) and concatenate it to the encoder feature. PNCC is the normalized mean shape of 3DMM Blanz & Vetter (2003), and is projected to fit a given face. It encodes the face semantics as each vertex (*e.g.*, eye corner) has its unique 3D coordinate between $[0, 0, 0]$ and $[1, 1, 1]$, regardless of the pose, expression and identity. Compared with conventional position encoding in Carion et al. (2020); Dosovitskiy et al. (2020), PNCC provides a better face semantic that helps to detect and remove shadows.

5.3.3 Colorization

An important takeaway from the grayscale shadow removal module is that we can locate the shadow region as

$$\hat{\mathbf{B}} = |\hat{\mathbf{I}}^{b,gs} - \mathbf{I}^{gs}| > \beta, \tag{5.7}$$

where $\hat{\mathbf{B}}$ is the shadow segmentation mask binarized with the threshold of β. With this knowledge, we can turn the blind color recovery process into an image inpainting process with a given inpainting region. In comparison, if no knowledge is provided to the colorization process, this two-step approach is nearly identical to direct RGB shadow removal applied in previous work Le & Samaras (2019); Shor & Lischinski (2008); Zhang et al. (2020b), which may still suffer from the poor generalization issue.

Our colorization module breaks down into 3 steps: 1) erasing, 2) inpainting, 3) color space transformation. Structurally, colorization module is similar with the grayscale shadow removal module. It consists of 3 residual non-local blocks and a decoder. First, based on the shadow mask $\hat{\mathbf{B}}$,

we set the shadow region of $\mathbf{F}$ as 0 to circumvent any potential disturbance, and term it as inpainting feature. Secondly, the inpainting feature $\mathbf{F} \odot (1 - \hat{\mathbf{B}})$ is concatenated with $\hat{\mathbf{B}}$ and the PNCC coding, and fed to the module. The non-local blocks aim to fill in the missing region in $\mathbf{F}$, and the decoder is designed to produce a M-channel color space $\mathbf{C} \in \mathbb{R}^{N^2 \times M}$. In the end, we use three 1×1 convolution layers to transfer grayscale face $\hat{\mathbf{I}}^{b,gs}$ with the color space $\mathbf{C}$ back to the RGB face $\hat{\mathbf{I}}^{b}$. During the training, no gradients of colorization module will be sent back to the grayscale shadow removal module via $\hat{\mathbf{B}}$.

5.3.4 Temporal information sharing

We can extend our network for single-frame process to leverage the temporal information via a Temporal Sharing Module (TSM). Similar to other video-based image restoration problems, such as video deblurring, shadow motion can be arbitrary in shape and speed variations. Thus, the order of the frames might not carry useful cues for de-shadow. As a result, we propose to adopt a temporal-wise max pooling to aggregate the illumination information among different frames, shown in Fig. 5.5.

Assuming $\mathbf{F}_1, \mathbf{F}_2, ..., \mathbf{F}_k$ are the features to be shared among k frames. Before computing the temporal-wise max pooling, we first apply a warping layer to register features based the face shape. After the temporal-wise max pooling, we apply a reverse warping to re-align the shared feature back to each frame feature, and concatenate with the original feature $\mathbf{F}_i$ for the next-stage computation. The TSM is a plug-in design for features at all scales. TSM can be used to not only share the temporal information, but also enforce the prior knowledge of face symmetry, which has been used in other tasks Wu et al. (2020). To achieve this, we treat the mirrored face as a different frame, and send to TSM for information sharing. In case there is only a single frame available, the TSM simply concatenates with the original feature.

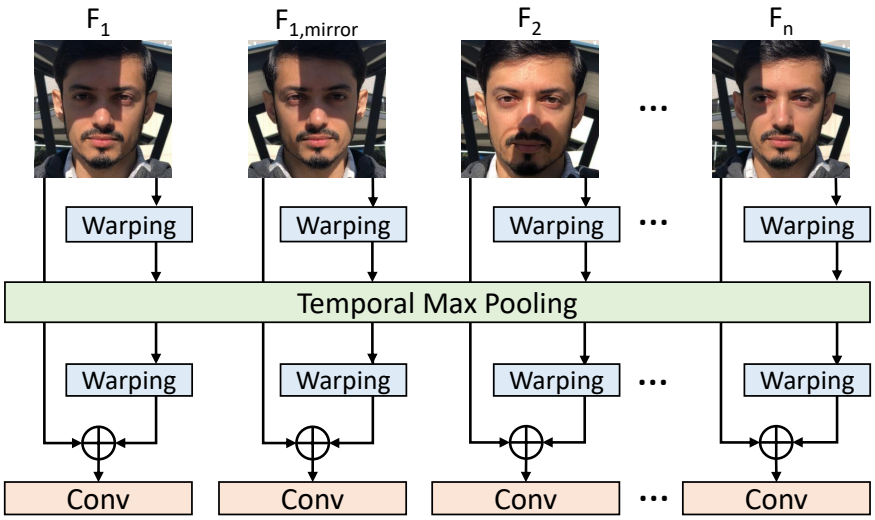


Figure 5.5 Illustration of the Temporal Sharing Module (TSM). It can be applied to temporal frames as well as mirrored input.

The warping layer leverages the pre-computed 68 facial landmarks via Bulat & Tzimiropoulos (2017). Given the landmarks for the neural face $\mathbf{s}_0$ and face $\mathbf{s}_i$ at the frame i, a sparse offset can be computed as $\Delta\mathbf{s}_{i\to 0} = \mathbf{s}_0 - \mathbf{s}_i \in \mathbb{R}^{68\times 2}$ to indicate where each pixel in the landmark position should be moved to. To obtain a dense offset map $\Delta\mathbf{S}_{i\to 0} \in \mathbb{R}^{N^2\times 2}$ indicating where each pixel in the entire feature map should be moved to, we apply a triangulation interpolation,

$$\Delta\mathbf{S}_{i\to 0} \leftarrow \text{Tri}(\mathbf{s}_i, \Delta\mathbf{s}_{i\to 0}, N), \tag{5.8}$$

where $\text{Tri}(\cdot)$ is Delaunay triangulation-based interpolation. The registration operation of feature $\mathbf{F}$ is denoted as:

$$\mathbf{F}_{i\to 0} = \mathbf{F}_i(\mathbf{S}^0 + \Delta\mathbf{S}_{i\to 0}), \tag{5.9}$$

where $\mathbf{S}_0 = \{(0,0),(0,1),...,(N,N)\} \in \mathbb{R}^{N^2\times 2}$ enumerates pixel locations in $\mathbf{F}_i$. Similarly, when we get the shared feature $\mathbf{F}_{\text{max}}$, we can use $\Delta\mathbf{S}_{0\to i}$ to warp it back.

5.3.5 Training

We use synthetic shadow faces to train our model. We apply multiple losses to supervise all three steps in the model.

Shadow removal loss: With the paired shadow face and well-illuminated face, we enable a pixel-to-pixel supervision on the recovery in grayscale. Specifically, we introduce a weighting map to encourage the loss to focus more on the shadow and shadow boundary, defined as

$$L_{gs} = \mathbb{E}_{i\sim P}\left[\left\|\hat{\mathbf{I}}_i^{b,gs} - \mathbf{I}_i^{b,gs}\right\|_1 \odot \frac{1+\mathbf{B}_i+\mathbf{B}_i^{edge}}{\mathbf{R}}\right], \tag{5.10}$$

where P indicates synthesized data distribution, $\mathbf{B}^{edge}$ is the boundary of $\mathbf{B}$, $1+\mathbf{B}+\mathbf{B}^{edge}$ is the weighting map, and $\mathbf{R}$ is the normalization term of the weighting map. A similar loss also applied to the final RGB recovery as L_{clr}.

Image gradient loss: Human vision is very sensitive to high frequency artifacts, such as edges. To further suppress artifacts around shadow boundaries and recover high-frequency details beneath shadows, we adopt an image gradient loss to encourage the image gradients between $\hat{\mathbf{I}}^b$ and $\mathbf{I}^b$ to be similar. This loss is denoted as:

$$\mathcal{L}_{\nabla} = \mathbb{E}_{i\sim P}\left[\sum_k \|\nabla\lfloor\hat{\mathbf{I}}_i^b\rfloor_k - \nabla\lfloor\mathbf{I}_i^b\rfloor_k\|_1\right], \tag{5.11}$$

where ∇ is the gradient operator and $\lfloor\cdot\rfloor_k$ denotes downsampling by the ratio $k=\{1,2,4,8\}$. Multiscale gradients help remove both sharp and blurring shadow boundaries.

Perceptual loss $\mathcal{L}_P$: To ensure the visual quality, we adopt the perceptual loss between the recovered face $\hat{\mathbf{I}}^b$ and $\mathbf{I}^b$.

GAN loss: Motivated by Wang et al. (2018b), we adopt a multiscale PatchGAN Isola et al. (2017)

at the scales 1, $1/2$, $1/4$ of the original image's resolution. Each discriminator consists of 5 convolutional layers and 4 pooling layers, and outputs a 1-channel map in the range of $[0, 1]$, where 0 denotes synthetic and 1 real. We use the hinge loss in the GAN training:

$$\mathcal{L}_D = -\mathbb{E}_{i\sim P}\left[\sum_{n=1,2,3}\min(0, -1 + D_n(\mathbf{I}_i^b))\right] - \mathbb{E}_{i\sim P}\left[\sum_{n=1,2,3}\min(0, -1 - D_n(\hat{\mathbf{I}}_i^b))\right],$$

$$\mathcal{L}_G = -\mathbb{E}_{i\sim P}\left[\sum_{n=1,2,3} D_n(\hat{\mathbf{I}}_i^b)\right], \tag{5.12}$$

where D_1, D_2 and D_3 are discriminators at 3 scales. $\mathcal{L}_D$ is the loss to train the discriminators and $\mathcal{L}_G$ is the loss to guide the shadow removal model to recover more realistic shadow-free faces.

Temporal consistency loss: We adopt a temporal consistency loss to encourage the image gradients between $\hat{\mathbf{I}}_i^{sf}$ and $\mathbf{I}_i^{sf}$ to be similar. This consistency loss is denoted as:

$$L_T = \mathbb{E}_{i\sim P}\left[\left\|\hat{\mathbf{I}}_{i,t_1}^b - \hat{\mathbf{I}}_{i,t_2}^b\right\|_1\right], \tag{5.13}$$

where t_1, t_2 can be either two nearby frames of the same video, nor the frame with its mirrored image.

Overall Loss The generator is supervised an overall loss as:

$$\mathcal{L} = \alpha_1\mathcal{L}_{gs} + \alpha_2\mathcal{L}_{clr} + \alpha_3\mathcal{L}_{\nabla} + \alpha_4\mathcal{L}_P + \alpha_5\mathcal{L}_G + \alpha_6\mathcal{L}_T. \tag{5.14}$$

The discriminators are supervised with adversarial loss $\mathcal{L}_D$ to compete with the generator. We execute the generator step and the discriminator step in each mini-batch iteration.

5.4 Training and Evaluation Data

Training data To synthesize our training data based on Eqn. 5.1-5.3, we manually select $15,000$ face images from FFHQ dataset Karras et al. (2019) that do not contain any foreign shadows and strong self shadows. The raw binary shadow shape $\mathbf{B}$ comes from: 1) 100 pre-defined silhouette shapes 2) Perlin noise function. After that, the raw shapes are randomly augmented with different scales, rotations and boundary blurriness. Intensity map $\mathbf{M}_I$ is also generated by random Perlin noise function at two octaves.

To simulate common shadow motion in face videos, we propose two approaches to synthesize the shadow motion: translation and flickering. In translation mode, the shadow moves on the face region with randomly selected speed, direction, and rotation. The shape of the shadow is fixed for each video but the scale, rotation, and boundary blurriness can be continuously shifting from frame to frame. In flickering mode, the location from frame to frame is randomly picked and the changes of scale, rotation and boundary blurriness are not continuous.

Evaluation data To our knowledge, there is no large video database of real-world human faces with foreign shadows. One existing database, UCB Zhang et al. (2020b), includes a very limited number of 100 face images. More importantly, this database contains only single images so that consistent image reconstruction on videos cannot be evaluated. In response to the need of a large video database, we collect a database termed Shadow Face in the Wild (SFW) for the evaluation of real-world facial shadow removal. In total, SFW includes 280 videos from 20 subjects. Some examples are shown in Fig. 5.6. Most videos are captured at 1080p resolution by various smartphone cameras.

For each subject, the videos are collected in five sessions: indoor, outdoor standing, outdoor walking, outdoor extreme, and driving. The indoor session collects face videos in an indoor

Figure 5.6 An illustration of SFW database. The first row shows the shadow faces collected under highly dynamic environments. (*e.g.*, varying shadows and head poses due to walking and driving); The second row shows the pixel-level annotations of shadow segmentation. Zoom in for viewing the quality of our annotation.

environment, where the lighting is relatively soft with no strong specular lights. For outdoor collection, the standing session requires the subject to hold a standing position with no ambient light variations, and the walking and extreme sessions require the subject to be moving, creating a changing ambient light. For the first three sessions, subjects use common objects to create shadows, such as hand, phone, paper, pen *etc*. For the outdoor extreme session, we strive to create more complex shadow patterns and require the subjects walking under trees to create leaf-shape shadows. In the last session, the subjects record videos in a moving car, where the shadows may come from the blocking of sun visor, rear-view mirror A-pillar, and surrounding buildings.

For the evaluation purpose, we annotate the pixel-level shadow segmentation maps of key frames selected from the video set, and more annotations will be added in the future.

5.5 Experiment

5.5.1 Experimental setup

Metrics To evaluate on UCB dataset, we can directly compare between the de-shadow face images from our model and the ground truth face images. We evaluate the performance based on the following metrics: peak signal-to-noise ratio (PSNR) and structural similarity index measure

Figure 5.7 A qualitative comparison of shadow removal on testing images of UCB database. From top to bottom, we show shadow face and shadow removal results provided by Zhang et al. (2020b), the network with naive RGB shadow modeling, our single-frame network with grayscale shadow removal and colorization (GS+C), and our network with additional TSM and temporal loss.

(SSIM). To evaluate on SFW dataset, we can evaluate on how the shadow is detected since the ground truth shadow segmentation is provided. The predicted shadow masks are from Eqn. 5.7. We compute the area under curve (AUC) of receiver operating characteristic curve (ROC) and accuracy based on the predicted shadow masks and the ground truth masks. The accuracy is computed as $\frac{TP+TN}{N_p+N_n}$ where TP, TN, N_p, and N_n are true positives, true negatives, number of shadow pixels, and number of non-shadow pixels, respectively. We binarize the shadow matte $\mathbf{M}$ into shadow mask with a threshold of 0.1.

Implementation details Our shadow removal network is implemented in Tensorflow with an initial learning rate of $1e$-4. We train the network for $50,000$ iterations in total with a batch size of 32, and decrease the learning rate by a ratio of 10 every $25,000$ iterations. We initialize the weights with the normal distribution of $[0, 0.02]$. $\{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6, \beta\}$ are set to be $\{100, 100, 1, 1, 1, 1, 0.1\}$. We use Bulat & Tzimiropoulos (2017) to crop the face and provide 68 facial landmarks.

Removal Model	PSNR	SSIM
Input Image	19.671	0.766
Guo *et al.*Guo et al. (2012)	15.939	0.593
Hu *et al.*Hu et al. (2018)	18.956	0.699
Cun *et al.*Cun et al. (2020)	19.386	0.722
Zhang *et al.*Zhang et al. (2020b)	**23.816**	**0.782**
Zhang *et al.*Zhang et al. (2020b)*	20.220	0.677
RGB	21.464	0.725
GS+C (Ours)	23.364	0.784
Temporal GS+C (Ours)	**23.793**	**0.805**

Table 5.1 A quantitative comparison for shadow removal on UCB dataset. Zhang *et al.* Zhang et al. (2020b)* is our implementation and trained using our synthesized data.

5.5.2 Shadow removal and segmentation

We first compare the results on UCB dataset. The baseline method is the one fromZhang et al. (2020b), which also includes the performance of several previous works Cun et al. (2020); Guo et al. (2012); Hu et al. (2018). However, as no pre-trained models, training data, and training scripts of all these methods are available, it is hard to obtain any fair comparison between these methods and ours on UCB. To bridge the gap, We re-implement Zhang et al. (2020b) with our best efforts, and train it using our synthesized data. We believe it to be a faithful implementation as the conventional perceptual loss and pixel-wise loss are mainly used. Table 5.1 reports the comparison results of PSNR and SSIM. Our single-frame grayscale shadow removal+colorization model (GS+C) outperforms the methods of Cun et al. (2020); Guo et al. (2012); Hu et al. (2018) and achieves comparable performance with Zhang et al. (2020b). With the temporal sharing module (TSM) and temporal consistency loss ($\mathcal{L}_T$), our method can achieve a competitive PSNR and outperform the reported Zhang et al. (2020b) on SSIM. Notice that both PSNR and SSIM from our implemented Zhang et al. (2020b) are lower than the reported numbers. We believe it is mainly due to the large domain gap between our training data and the training data used in Zhang et al. (2020b). A qualitative comparison is shown in Fig. 5.7.

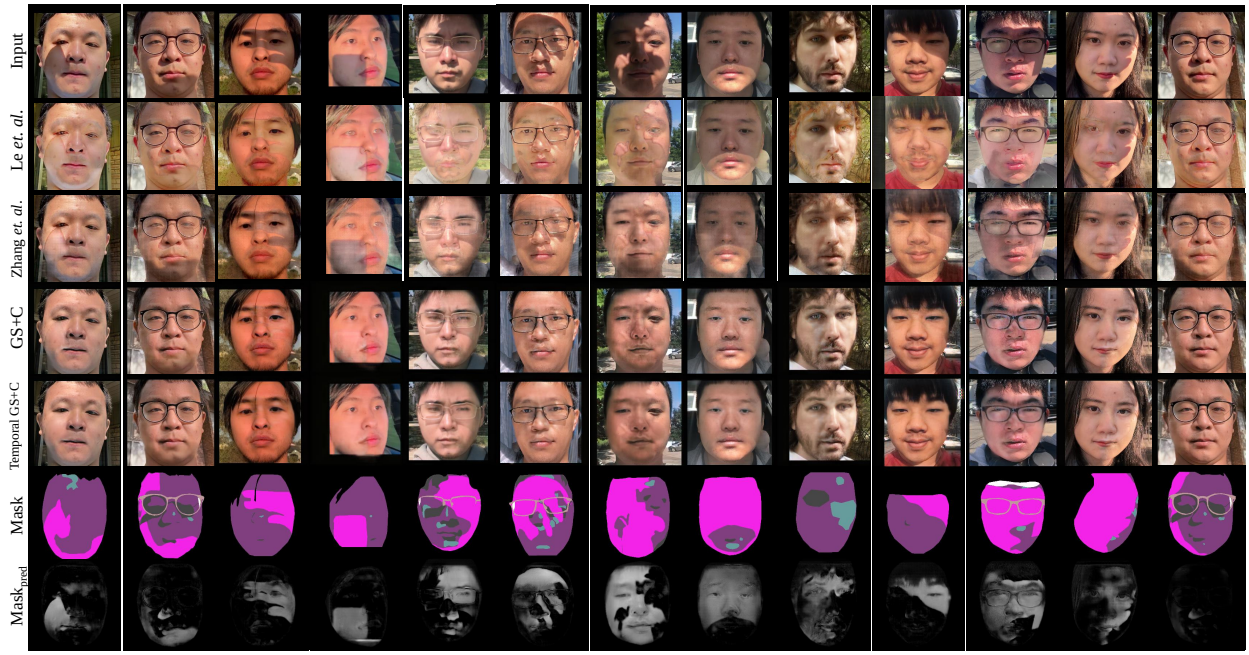


Figure 5.8 Qualitative shadow removal evaluations on SFW database. From top to bottom, we show shadow face, shadow removal results from Le & Samaras (2019) and Zhang et al. (2020b), our single-frame model, and our temporal model, ground truth shadow segmentation (in bright purple), and predicted shadow mask (before thresholding).

Secondly, we evaluate the models on the SFW database, which is more challenging due to highly dynamic environments. We conduct a quantitative comparison on the performance of shadow segmentation, which is a required module in many applications. Table 5.2 shows the comparison with recent methods Hu et al. (2019); Le & Samaras (2019); Zhang et al. (2020b). Our method outperforms all others in terms of AUC and accuracy. Fig. 5.8 shows the qualitative comparison and our method produces much better recovery of facial foreign shadows comparing with the baselines. Furthermore, our model is able to remove strong self-shadows (the forehead area of the 9th column) and keep normal shadows (the left-cheek area of the last column). As our datasets are highly diverse, we find that methods in Le & Samaras (2019); Zhang et al. (2020b) cannot be well generalized and their performance degrades.

Segmentation Model	AUC	Accuracy
Le *et al.*Le & Samaras (2019)	0.603	0.683
Hu *et al.*Hu et al. (2019)	0.540	0.604
Zhang *et al.*Zhang et al. (2020b)	0.686	0.756
GS+C	0.898	0.874
Temporal GS+C	**0.904**	**0.882**

Table 5.2 A quantitative comparison of shadow segmentation on SFW database.

5.5.3 Ablation Studies

We conduct ablation to better understand each component. The baseline method is our implemented version of Zhang et al. (2020b) and the direct RGB shadow modeling with our backbone network. For a fair comparison, we merge the computation resource of GS+C model (*i.e.*, doubling the bottleneck depth and the decoder channel.). As shown in Tab. 5.1, our implemented Zhang et al. (2020b) shows a performance of PSNR as 20.220 and SSIM as 0.677. By updating to a better backbone network with non-local blocks and face position coding, our baseline model with RGB shadow modeling achieves a better PSNR of 21.464 and SSIM of 0.725. Next, our single-frame GS+C model outperforms the previous two baseline models, thanks to the effectiveness of novel shadow modeling. And with the temporal design of TSM and the corresponding loss, our model can further improve the PSNR and SSIM to 23.793 and 0.805 respectively. Qualitative comparison are shown in Fig. 5.7. We can see both single-frame and temporal GS+C model show better visual quality to the RGB model. In addition, the temporal model further improves the color consistency and suppresses the artifacts on several subjects.

5.6 Conclusion

In this chapter, we introduce the problem of blind removal of facial foreign shadow. We propose an effective shadow modeling to help the model to be more generalized. We decompose the

conventional RGB shadow modeling into grayscale shadow modeling and colorization. We also propose a temporal sharing module (TSM) that can be easily integrated into any encoders and decoders to impose temporal consistency. Our method can produce photo-realistic de-shadow faces with high PSNR and SSIM. Our large-scale video database collected under highly dynamic environments is another major contribution that can benefit various face-related researches and applications.

Chapter 6

Conclusions and Future Work

Face is one of the most popular biometric modalities due to its convenience of usage, e.g., access control, phone unlock. Despite the high recognition accuracy, face recognition systems are not able to distinguish between real human faces and fake ones. Thus, they are vulnerable to face spoof attacks, which deceives the systems to recognize as another person. To safely use face recognition, face anti-spoofing techniques are required to detect spoof attacks before performing recognition.

In Chapter 2, we propose a CNN-RNN model is learned to estimate the face depth with pixel-wise supervision, and to estimate rPPG signals with sequence-wise supervision. The estimated depth and rPPG are fused to distinguish live vs. spoof faces. Experiments show that our model achieves significant improvements on both intra- and cross-database detection performance.

In Chapter 3, we study the generalization problem of face anti-spoofing. We define the detection of unknown spoof attacks as Zero-Shot Face Anti-spoofing (ZSFA) and extend the study of ZSFA from 1-2 types to 13 types. We propose a novel Deep Tree Network (DTN) to partition the spoof samples into semantic sub-groups in an unsupervised fashion. Experiments show that our proposed method achieves the state of the art on multiple testing protocols of ZSFA.

In Chapter 4, we study a new problem of interpreting face anti-spoofing model's decision. We provide a comprehensive modeling of the spoof traces of various spoof attacks, and designs a novel adversarial learning framework to disentangle the spoof traces from input faces as a hierarchical combination of patterns at multiple scales. With the disentangled spoof traces, we unveil the live counterpart of the original spoof face, and further synthesize realistic new spoof faces after a proper

geometric correction. Our method demonstrates superior spoof detection performance on both seen and unseen spoof scenarios while providing visually-convincing estimation of spoof traces.

In Chapter 5, we find the proper physical modeling can also benefit other face problems and study a new problem of face shadow removal. We propose an effective shadow modeling to help the model to be more generalized.We decompose the conventional RGB shadow modeling into grayscale shadow modeling and colorization. In addition, we propose a temporal sharing module (TSM) that can be easily integrated into any encoders and decoders to impose temporal consistency.

6.1 Future Works

Uncertainty of face anti-spoofing We look into open-set face anti-spoofing, and we find sometime we find the model may not be always very confident about its decision. In practice, sometime it's okay to reject the sample or provide additional manual inspection when the model is not very confident. So a future direction is to enable the model to provide a confidence score with its decision.

Retrainability of face anti-spoofing In practice, the face anti-spoofing model may need to deliver to different users to handle different situation, where the finetuning process, or retraining process is engaged. To ease the retraining process, several topcis can be further invastigated, such as incremental learning, model fusion and early stopping policy.

Sensor variations Sensor variation is an important factor in practical face anti-spoofing system, which haven't been quantitatively studied. While sensor variation causes large negative impact on the face anti-spoofing performance, face anti-spoofing models have to be re-trained every time switching to a new sensor, which is merely possible and consuming. We intend to first evaluate the cross-sensor performance and propose solution if the performance is not ideal.

Improving synthesis on large pose and expression We intend to modify the 3D Warping Layer

to better handle large pose variation. Specifically, we additionally provide visibility in the landmark preparation, so the warping layer can leverage the visibility to implement a fast and differentiable z-buffer rendering. This can additionally leverage other large pose face databases, such as 300-VW, to synthesize large pose spoof faces for training, which is hard to include in real face anti-spoofing databases.

APPENDIX

Contribution on Co-authored Publications

Chapters 2 of this dissertation are based on research papers Liu et al. (2018c) and Jourabloo et al. (2018) respectively. Amin Jourabloo and I have equal contributions on the proposed methods and implementation of these research papers. Here, I mention the detail contributions of each individual:

1. Learning Deep Models for Face Anti-Spoofing: Binary or Auxiliary Supervision [Liu et al. (2018c)]

Yaojie Liu:

◇ Data collection for SiW dataset;

◇ Providing the ground truth for the pseudo-depth map for the training images;

◇ Designing the CNN part of the network, including Depth CNN and non-rigid registration layer;

◇ Implementing the training on different datasets and protocols, and finalizing training settings and tricks;

◇ Carrying on the daily management of SiW dataset, including grant accesses to research groups worldwide, and answering questions of the database.

Amin Jourabloo:

◇ Data collection for SiW dataset;

◇ Providing the ground truth rPPG signals for the training videos;

◇ Designing the RNN part of the network, including the LSTM, FFT layer, and corresponding loss function;

◇ Implementing evaluation metrics (such as EER, HTER, and ACER), performing testing on

different datasets and protocols, and generate qualitative and quantitative results and analysis.

2. Face De-Spoofing: Anti-Spoofing via Noise Modeling [Jourabloo et al. (2018)]

Yaojie Liu:

◇ Performing a case study on spoof noise pattern, and explore three important properties of spoof noise pattern;

◇ Implementing experiments and analyzing results on different datasets and protocols (CASIA and Replay Attack datasets);

Amin Jourabloo:

◇ Designing the loss functions and the network architecture for the face de-spoofing, including the DS Net, DQ Net, and VQ Net;

◇ Implementing the training on all protocols in Oulu database, and finalizing training settings and tricks;

◇ Analyzing the experiment results on Oulu dataset and executing several ablation studies.

Related Publication

In this section, I list all related publication I finished during my PhD study in Michigan State University:

1. Liu, Xiaohong, Yaojie Liu, Jun Chen & Xiaoming Liu. 2021. PSCC-Net: Progressive spatio-channel correlation network for image manipulation detection and localization. *arXiv preprint arXiv:2103.10596.*

2. Liu, Yaojie & Xiaoming Liu, Physics-Guided Spoof Trace Disentanglement for Generic Face Anti-Spoofing. *arXiv preprint arXiv:2012.05185.*

3. Liu, Yaojie, Joel Stehouwer & Xiaoming Liu. 2020. On Disentangling Spoof Traces for Generic Face Anti-spoofing. In *ECCV*.

4. Stehouwer, Joel, Amin Jourabloo, Yaojie Liu & Xiaoming Liu. 2020. Noise Modeling, Synthesis and Classification for Generic Object Anti-spoofing. In *CVPR*, IEEE.

5. Liu, Yaojie, Joel Stehouwer, Amin Jourabloo & Xiaoming Liu. 2019. Presentation Attack Detection for Face in Mobile Phones, In *Selfie Biometrics*, Springer.

6. Liu, Yaojie, Joel Stehouwer, Amin Jourabloo & Xiaoming Liu. 2019. Deep Tree Learning for Zero-shot Face Anti-spoofing. In *CVPR*, IEEE.

7. Jourabloo, Amin*, Yaojie Liu* & Xiaoming Liu. 2018. Face De-spoofing: Anti-spoofing via Noise Modeling. In *ECCV*.

8. Liu, Yaojie*, Amin Jourabloo* & Xiaoming Liu. 2018. Learning Deep Models for Face Anti-spoofing: Binary or Auxiliary Supervision. In *CVPR*, IEEE.

9. Atoum, Yousef*, Yaojie Liu*, Amin Jourabloo* & Xiaoming Liu. 2017. Face Anti-spoofing Using Patch and Depth-based CNNs. In *IJCB*, IEEE.

10. Liu, Yaojie, Amin Jourabloo, William Ren & Xiaoming Liu. 2017. Dense Face Alignment. In *ICCVW*, IEEE.

11. Liu, Yaojie, Xinyu Huang, Liu Ren & Xiaoming Liu. 2021. Blind Removal of Facial Foreign Shadow. Submitted to *ICCV*.

BIBLIOGRAPHY

BIBLIOGRAPHY

Abadi, Martín, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard et al. 2016. Tensorflow: A system for large-scale machine learning. In *OSDI*, .

Abdelhamed, Abdelrahman, Stephen Lin & Michael S Brown. 2018. A high-quality denoising dataset for smartphone cameras. In *CVPR*, IEEE.

Agarwal, Akshay, Richa Singh & Mayank Vatsa. 2016. Face anti-spoofing using Haralick features. In *BTAS*, IEEE.

Arashloo, Shervin Rahimzadeh, Josef Kittler & William Christmas. 2017. An anomaly detection approach to face spoofing detection: A new formulation and evaluation protocol. *IEEE Access* .

Arrieta, Alejandro Barredo, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins et al. 2020. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion* .

Atoum, Yousef, Yaojie Liu, Amin Jourabloo & Xiaoming Liu. 2017. Face anti-spoofing using patch and depth-based CNNs. In *IJCB*, IEEE.

Bao, Wei, Hong Li, Nan Li & Wei Jiang. 2009. A liveness detection method for face recognition based on optical flow field. In *IASP*, .

Bertalmio, Marcelo, Guillermo Sapiro, Vincent Caselles & Coloma Ballester. 2000. Image inpainting. In *Proceedings of the 27th annual conference on computer graphics and interactive techniques*, .

Bharadwaj, Samarth, Tejas I Dhamecha, Mayank Vatsa & Richa Singh. 2013. Computationally efficient face spoofing detection with motion magnification. In *CVPRW*, IEEE.

Bharadwaj, Samarth, Tejas I Dhamecha, Mayank Vatsa & Richa Singh. 2014. Face anti-spoofing via motion magnification and multifeature videolet aggregation .

Blanz, Volker & Thomas Vetter. 2003. Face recognition based on fitting a 3D morphable model. *PAMI* .

Bobbia, Serge, Yannick Benezeth & Julien Dubois. 2016. Remote photoplethysmography based on implicit living skin tissue segmentation. In *ICPR*, .

Boulkenafet, Zinelabdine. 2017. A competition on generalized software-based face presentation attack detection in mobile scenarios. In *IJCB*, IEEE.

Boulkenafet, Zinelabidine, Jukka Komulainen & Abdenour Hadid. 2015. Face anti-spoofing based on color texture analysis. In *ICIP*, IEEE.

Boulkenafet, Zinelabidine, Jukka Komulainen & Abdenour Hadid. 2016. Face spoofing detection using colour texture analysis. *TIFS* .

Boulkenafet, Zinelabidine, Jukka Komulainen & Abdenour Hadid. 2017a. Face antispoofing using speeded-up robust features and Fisher vector encoding. *Signal Process Letters* .

Boulkenafet, Zinelabinde, Jukka Komulainen, Lei Li, Xiaoyi Feng & Abdenour Hadid. 2017b. OULU-NPU: A mobile face presentation attack database with real-world variations. In *FG*, .

Bulat, Adrian & Georgios Tzimiropoulos. 2017. How far are we from solving the 2d & 3d face alignment problem?(and a dataset of 230,000 3d facial landmarks). In *ICCV*, IEEE.

Cao, Chen, Yanlin Weng, Shun Zhou, Yiying Tong & Kun Zhou. 2014. Facewarehouse: A 3D facial expression database for visual computing. *Trans. Vis. Comput. Graphics* .

Cao, Qingxing, Xiaodan Liang, Bailing Li, Guanbin Li & Liang Lin. 2018. Visual question reasoning on general dependency tree. In *CVPR*, IEEE.

Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov & Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In *ECCV*, Springer.

Chang, Huiwen, Jingwan Lu, Fisher Yu & Adam Finkelstein. 2018. PairedCycleGAN: Asymmetric style transfer for applying and removing makeup. In *CVPR*, IEEE.

Chen, Chang, Zhiwei Xiong, Xiaoming Liu & Feng Wu. 2020. Camera trace erasing. In *CVPR*, IEEE.

Chen, Cunjian, Antitza Dantcheva & Arun Ross. 2013. Automatic facial makeup detection with application in face recognition. In *ICB*, IEEE.

Chen, Cunjian, Antitza Dantcheva & Arun Ross. 2014. Impact of facial cosmetics on automatic gender and age estimation algorithms. In *International conference on computer vision theory and applications (visapp)*, IEEE.

Chen, Xinyun, Chang Liu & Dawn Song. 2018. Tree-to-tree neural networks for program translation. In *NIPS*, .

Chetty, Girija. 2010. Biometric liveness checking using multimodal fuzzy fusion. In *International conference on fuzzy systems*, IEEE.

Chetty, Girija & Michael Wagner. 2006. Audio-visual multimodal fusion for biometric person authentication and liveness verification. In *Proceedings of the 2005 nicta-hcsnet multimodal user interaction workshop-volume 57*, .

Chingovska, Ivana, André Anjos & Sébastien Marcel. 2012. On the effectiveness of local binary patterns in face anti-spoofing. In *BIOSIG*, IEEE.

Chuang, Yung-Yu, Dan B Goldman, Brian Curless, David H Salesin & Richard Szeliski. 2003. Shadow matting and compositing. In *Siggraph*, ACM.

Corneanu, Ciprian A, Meysam Madadi, Sergio Escalera & Aleix M Martinez. 2019. What does it mean to learn in deep networks? and, how does one detect adversarial attacks? In *CVPR*, IEEE.

Cun, Xiaodong, Chi-Man Pun & Cheng Shi. 2020. Towards ghost-free shadow removal via dual hierarchical aggregation network and shadow matting GAN. In *AAAI*, .

Dang, Hao, Feng Liu, Joel Stehouwer, Xiaoming Liu & Anil K Jain. 2020. On the detection of digital face manipulation. In *CVPR*, IEEE.

Ding, Bin, Chengjiang Long, Ling Zhang & Chunxia Xiao. 2019. ARGAN: Attentive recurrent generative adversarial network for shadow detection and removal. In *ICCV*, IEEE.

Donner, Craig & Henrik Wann Jensen. 2006. A spectral BSSRDF for shading human skin. In *Proceedings of the 17th eurographics conference on rendering techniques*, 409–417.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* .

Egger, Bernhard, Sandro Schönborn, Andreas Schneider, Adam Kortylewski, Andreas Morel-Forster, Clemens Blumer & Thomas Vetter. 2018. Occlusion-aware 3D morphable models and an illumination prior for face image analysis. *IJCV* .

Esser, Patrick, Ekaterina Sutter & Björn Ommer. 2018. A variational U-Net for conditional appearance and shape generation. In *CVPR*, IEEE.

Feng, Haocheng, Zhibin Hong, Haixiao Yue, Yang Chen, Keyao Wang, Junyu Han, Jingtuo Liu & Errui Ding. 2020. Learning generalized spoof cues for face anti-spoofing. *arXiv preprint arXiv:2005.03922* .

Feng, Litong, Lai-Man Po, Yuming Li, Xuyuan Xu, Fang Yuan, Terence Chun-Ho Cheung & Kwok-Wai Cheung. 2016. Integration of image quality and motion cues for face anti-spoofing: A neural network approach. *J. Visual Communication and Image Representation* .

Finlayson, Graham D, Steven D Hordley & Mark S Drew. 2002. Removing shadows from images. In *ECCV*, Springer.

de Freitas Pereira, Tiago, André Anjos, José Mario De Martino & Sébastien Marcel. 2012. LBP-TOP based countermeasure against face spoofing attacks. In *ACCV*, IEEE.

de Freitas Pereira, Tiago, André Anjos, José Mario De Martino & Sébastien Marcel. 2013. Can face anti-spoofing countermeasures work in a real world scenario? In *ICB*, IEEE.

Frome, Andrea, Greg S Corrado, Jon Shlens, Samy Bengio, Jeff Dean, Marc'Aurelio Ranzato & Tomas Mikolov. 2013. Devise: A deep visual-semantic embedding model. In *NIPS*, .

Gu, Shuyang, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen & Lu Yuan. 2019. Mask-guided portrait editing with conditional GANs. In *CVPR*, IEEE.

Guo, Jianzhu, Xiangyu Zhu, Jinchuan Xiao, Zhen Lei, Genxun Wan & Stan Z Li. 2019. Improving face anti-spoofing by 3D virtual synthesis. *arXiv preprint arXiv:1901.00488* .

Guo, Ruiqi, Qieyun Dai & Derek Hoiem. 2012. Paired regions for shadow detection and removal. *PAMI* .

de Haan, Gerard & Vincent Jeanne. 2013. Robust pulse rate from chrominance-based rPPG. *Trans. Biomedical Engineering* .

He, Kaiming, Xiangyu Zhang, Shaoqing Ren & Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*, IEEE.

Hu, X, CW Fu, L Zhu, J Qin & PA Heng. 2019. Direction-aware spatial context features for shadow detection and removal. *PAMI* .

Hu, Xiaowei, Lei Zhu, Chi-Wing Fu, Jing Qin & Pheng-Ann Heng. 2018. Direction-aware spatial context features for shadow detection. In *CVPR*, IEEE.

IARPA. 2016. IARPA research program Odin. `https://www.iarpa.gov/index.php/research-programs/odin`.

ISO/IEC-JTC-1/SC-37. 2016. Biometrics. information technology biometric presentation attack detection part 1: Framework. `https://www.iso.org/obp/ui/iso`.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou & Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *CVPR*, IEEE.

Jourabloo, Amin & Xiaoming Liu. 2017. Pose-invariant face alignment via CNN-based dense 3D model fitting. *IJCV* .

Jourabloo, Amin, Yaojie Liu & Xiaoming Liu. 2018. Face de-spoofing: Anti-spoofing via noise modeling. In *ECCV*, .

Kalchbrenner, Nal, Edward Grefenstette & Phil Blunsom. 2014. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188* .

Kaneko, Takuhiro, Kaoru Hiramatsu & Kunio Kashino. 2018. Generative adversarial image synthesis with decision tree latent controller. In *CVPR*, IEEE.

Karessli, Nour, Zeynep Akata, Bernt Schiele & Andreas Bulling. 2017. Gaze embeddings for zero-shot image classification. In *CVPR*, IEEE.

Karras, Tero, Samuli Laine & Timo Aila. 2019. A style-based generator architecture for generative adversarial networks. In *CVPR*, IEEE.

Kazemi, Vahid & Josephine Sullivan. 2014. One millisecond face alignment with an ensemble of regression trees. In *CVPR*, IEEE.

Kollreider, Klaus, Hartwig Fronthaler, Maycel Isaac Faraj & Josef Bigun. 2007. Real-time face detection and motion analysis with application in liveness assessment. *TIFS* .

Komulainen, Jukka, Abdenour Hadid & Matti Pietikainen. 2013a. Context based face anti-spoofing. In *BTAS*, IEEE.

Komulainen, Jukka, Abdenour Hadid, Matti Pietikäinen, André Anjos & Sébastien Marcel. 2013b. Complementary countermeasures for detecting scenic face spoofing attacks. In *ICB*, IEEE.

Krizhevsky, Alex, Ilya Sutskever & Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *NIPS*, .

Lampert, Christoph H, Hannes Nickisch & Stefan Harmeling. 2009. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR*, IEEE.

Lawrence, Steve, C Lee Giles, Ah Chung Tsoi & Andrew D Back. 1997. Face recognition: A convolutional neural-network approach. *IEEE Transactions on Neural Networks* .

Le, Hieu & Dimitris Samaras. 2019. Shadow removal via shadow image decomposition. In *ICCV*, IEEE.

Lee, Cheng-Han, Ziwei Liu, Lingyun Wu & Ping Luo. 2020. MaskGAN: Towards diverse and interactive facial image manipulation. In *CVPR*, IEEE.

Li, Jiangwei, Yunhong Wang, Tieniu Tan & Anil K Jain. 2004. Live face detection based on the analysis of fourier spectra. In *BTHI*, SPIE.

Li, Lei, Xiaoyi Feng, Zinelabidine Boulkenafet, Zhaoqiang Xia, Mingming Li & Abdenour Hadid. 2016a. An original face anti-spoofing approach using partial convolutional neural network. In *IPTA*, .

Li, Xiaobai, Jukka Komulainen, Guoying Zhao, Pong-Chi Yuen & Matti Pietikäinen. 2016b. Generalized face anti-spoofing by detecting pulse from face videos. In *ICPR*, IEEE.

Li, Zhihang, Yibo Hu, Ran He & Zhenan Sun. 2020. Learning disentangling and fusing networks for face completion under structured occlusions. *Pattern Recognition* .

Liao, Jing, Yuan Yao, Lu Yuan, Gang Hua & Sing Bing Kang. 2017. Visual attribute transfer through deep image analogy. *arXiv preprint arXiv:1705.01088* .

Liu, Feng, Dan Zeng, Qijun Zhao & Xiaoming Liu. 2018a. Disentangling features in 3D face shapes for joint face reconstruction and recognition. In *CVPR*, IEEE.

Liu, Si-Qi, Xiangyuan Lan & Pong C Yuen. 2018b. Remote photoplethysmography correspondence feature for 3d mask face presentation attack detection. In *ECCV*, 558–573.

Liu, Siqi, Baoyao Yang, Pong C Yuen & Guoying Zhao. 2016a. A 3D mask face anti-spoofing database with real world variations. In *CVPRW*, IEEE.

Liu, Siqi, Pong C Yuen, Shengping Zhang & Guoying Zhao. 2016b. 3D mask face anti-spoofing with remote photoplethysmography. In *ECCV*, .

Liu, Yaojie, Amin Jourabloo & Xiaoming Liu. 2018c. Learning deep models for face anti-spoofing: Binary or auxiliary supervision. In *CVPR*, IEEE.

Liu, Yaojie, Amin Jourabloo, William Ren & Xiaoming Liu. 2017. Dense face alignment. In *ICCVW*, IEEE.

Liu, Yaojie & Chang Shu. 2015. A comparison of image inpainting techniques. In *Sixth international conference on graphic and image processing (icgip)*, International Society for Optics and Photonics.

Liu, Yaojie, Joel Stehouwer, Amin Jourabloo & Xiaoming Liu. 2019a. Deep tree learning for zero-shot face anti-spoofing. In *CVPR*, IEEE.

Liu, Yaojie, Joel Stehouwer, Amin Jourabloo & Xiaoming Liu. 2019b. Presentation attack detection for face in mobile phones. *Selfie Biometrics* .

Liu, Yaojie, Joel Stehouwer & Xiaoming Liu. 2020. On disentangling spoof traces for generic face anti-spoofing. In *ECCV*, .

Maaten, Laurens van der & Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* .

Määttä, Jukka, Abdenour Hadid & Matti Pietikäinen. 2011. Face spoofing detection from single images using micro-texture analysis. In *IJCB*, IEEE.

Mao, Xudong, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang & Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *ICCV*, IEEE.

Mirjalili, Vahid & Arun Ross. 2017. Soft biometric privacy: Retaining biometric utility of face images while perturbing gender. In *ICB*, IEEE.

Nagano, Koki, Huiwen Luo, Zejian Wang, Jaewoo Seo, Jun Xing, Liwen Hu, Lingyu Wei & Hao Li. 2019. Deep face normalization. *TOG* .

Nestmeyer, Thomas, Jean-François Lalonde, Iain Matthews & Andreas Lehrmann. 2020. Learning physics-guided face relighting under directional light. In *CVPR*, IEEE.

Nowara, Ewa Magdalena, Ashutosh Sabharwal & Ashok Veeraraghavan. 2017. Ppgsecure: Biometric presentation attack detection using photopletysmograms. In *FG*, .

Pan, Gang, Lin Sun, Zhaohui Wu & Shihong Lao. 2007. Eyeblink-based anti-spoofing in face recognition from a generic webcamera. In *ICCV*, IEEE.

Patel, Keyurkumar, Hu Han & Anil K Jain. 2016a. Cross-database face antispoofing with robust feature representation. In *CCBR*, .

Patel, Keyurkumar, Hu Han & Anil K Jain. 2016b. Secure face unlock: Spoof detection on smartphones. *TIFS* .

Paysan, Pascal, Reinhard Knothe, Brian Amberg, Sami Romdhani & Thomas Vetter. 2009. A 3D face model for pose and illumination invariant face recognition. In *AVSS*, .

Peixoto, Bruno, Carolina Michelassi & Anderson Rocha. 2011. Face liveness detection under bad illumination conditions. In *ICIP*, IEEE.

Pinto, Allan, Helio Pedrini, William Robson Schwartz & Anderson Rocha. 2015. Face spoofing detection through visual codebooks of spectral temporal cubes. *TIP* .

Po, Lai-Man, Litong Feng, Yuming Li, Xuyuan Xu, Terence Chun-Ho Cheung & Kwok-Wai Cheung. 2017. Block-based adaptive ROI for remote photoplethysmography. *J. Multimedia Tools and Applications* .

Qin, Yunxiao, Chenxu Zhao, Xiangyu Zhu, Zezheng Wang, Zitong Yu, Tianyu Fu, Feng Zhou, Jingping Shi & Zhen Lei. 2019. Learning meta model for zero-and few-shot face anti-spoofing. *arXiv preprint arXiv:1904.12490* .

Qu, Liangqiong, Jiandong Tian, Shengfeng He, Yandong Tang & Rynson WH Lau. 2017. Deshadownet: A multi-context embedding deep network for shadow removal. In *CVPR*, IEEE.

Ronneberger, Olaf, Philipp Fischer & Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *International conference on medical image computing and computer-assisted intervention*, Springer.

Sengupta, Soumyadip, Angjoo Kanazawa, Carlos D Castillo & David W Jacobs. 2018. SfSNet: Learning shape, reflectance and illuminance of facesin the wild'. In *CVPR*, IEEE.

Shao, Rui, Xiangyuan Lan, Jiawei Li & Pong C Yuen. 2019a. Multi-adversarial discriminative deep domain generalization for face presentation attack detection. In *CVPR*, IEEE.

Shao, Rui, Xiangyuan Lan & Pong C. Yuen. 2017. Deep convolutional dynamic texture learning with adaptive channel-discriminability for 3D mask face anti-spoofing. In *IJCB*, IEEE.

Shao, Rui, Xiangyuan Lan & Pong C Yuen. 2019b. Regularized fine-grained meta face anti-spoofing. *arXiv preprint arXiv:1911.10771* .

Shashua, Amnon & Tammy Riklin-Raviv. 2001. The quotient image: Class-based re-rendering and recognition with varying illuminations. *PAMI* .

Shih, YiChang, Sylvain Paris, Connelly Barnes, William T Freeman & Frédo Durand. 2014. Style transfer for headshot portraits .

Shor, Yael & Dani Lischinski. 2008. The shadow meets the mask: Pyramid-based shadow removal. In *Computer graphics forum*, Wiley Online Library.

Shu, Zhixin, Sunil Hadap, Eli Shechtman, Kalyan Sunkavalli, Sylvain Paris & Dimitris Samaras. 2017a. Portrait lighting transfer using a mass transport approach. *TOG* .

Shu, Zhixin, Ersin Yumer, Sunil Hadap, Kalyan Sunkavalli, Eli Shechtman & Dimitris Samaras. 2017b. Neural face editing with intrinsic image disentangling. In *CVPR*, IEEE.

Socher, Richard, Milind Ganjoo, Christopher D Manning & Andrew Ng. 2013. Zero-shot learning through cross-modal transfer. In *NIPS*, .

Stehouwer, Joel, Amin Jourabloo, Yaojie Liu & Xiaoming Liu. 2020. Noise modeling, synthesis and classification for generic object anti-spoofing. In *CVPR*, IEEE.

Stoschek, Arne. 2000. Image-based re-rendering of faces for continuous pose and illumination directions. In *CVPR*, IEEE.

Sun, Lin, Gang Pan, Zhaohui Wu & Shihong Lao. 2007. Blinking-based live face detection using conditional random fields. *J. Advances in Biometrics* .

Sun, Tiancheng, Jonathan T Barron, Yun-Ta Tsai, Zexiang Xu, Xueming Yu, Graham Fyffe, Christoph Rhemann, Jay Busch, Paul E Debevec & Ravi Ramamoorthi. 2019. Single image portrait relighting. *TOG* .

Tan, Xiaoyang, Yi Li, Jun Liu & Lin Jiang. 2010. Face liveness detection from a single image with sparse low rank bilinear discriminative model. In *ECCV*, .

Tewari, Ayush, Michael Zollhofer, Hyeongwoo Kim, Pablo Garrido, Florian Bernard, Patrick Perez & Christian Theobalt. 2017. MoFA: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *ICCVW*, IEEE.

Thai, Thanh Hai, Remi Cogranne & Florent Retraint. 2013. Camera model identification based on the heteroscedastic noise model. *TIP* .

Thai, Thanh Hai, Florent Retraint & Rémi Cogranne. 2016. Camera model identification based on the generalized noise model in natural images. *Digital Signal Processing* .

Tran, Luan & Xiaoming Liu. 2018. Nonlinear 3D face morphable model. In *CVPR*, IEEE.

Tran, Luan & Xiaoming Liu. 2021. On learning 3d face morphable model from in-the-wild images. *PAMI* .

Tran, Luan, Xiaoming Liu, Jiayu Zhou & Rong Jin. 2017a. Missing modalities imputation via cascaded residual autoencoder. In *CVPR*, IEEE.

Tran, Luan, Xi Yin & Xiaoming Liu. 2017b. Disentangled representation learning GAN for pose-invariant face recognition. In *CVPR*, IEEE.

Tulyakov, Sergey, Xavier Alameda-Pineda, Elisa Ricci, Lijun Yin, Jeffrey F Cohn & Nicu Sebe. 2016. Self-adaptive matrix completion for heart rate estimation from face videos under realistic conditions. In *CVPR*, IEEE.

Turek, Matt. 2016. Explainable Artificial Intelligence (XAI). `https://www.darpa.mil/program/explainable-artificial-intelligence`.

Valle, Roberto, Jose M Buenaposada, Antonio Valdes & Luis Baumela. 2018. A deeply-initialized coarse-to-fine ensemble of regression trees for face alignment. In *ECCV*, .

Wang, Jifeng, Xiang Li & Jian Yang. 2018a. Stacked conditional generative adversarial networks for jointly learning shadow detection and shadow removal. In *CVPR*, IEEE.

Wang, Ting-Chun, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz & Bryan Catanzaro. 2018b. High-resolution image synthesis and semantic manipulation with conditional GANs. In *CVPR*, IEEE.

Wang, Xiaolong, Ross Girshick, Abhinav Gupta & Kaiming He. 2018c. Non-local neural networks. In *CVPR*, IEEE.

Wang, Yang, Lei Zhang, Zicheng Liu, Gang Hua, Zhen Wen, Zhengyou Zhang & Dimitris Samaras. 2008. Face relighting from a single image under arbitrary unknown lighting conditions. *PAMI* .

Wang, Yu, Luca Bondi, Paolo Bestagini, Stefano Tubaro, David J Edward Delp et al. 2017. A counter-forensic method for cnn-based camera model identification. In *CVPRW*, IEEE.

Wen, Di, Hu Han & A.K. Jain. 2015. Face spoof detection with image distortion analysis. *TIFS* .

Wen, Zhen, Zicheng Liu & Thomas S Huang. 2003. Face relighting with radiance environment maps. In *CVPR*, IEEE.

Wu, Bing-Fei, Yun-Wei Chu, Po-Wei Huang, Meng-Liang Chung & Tzu-Min Lin. 2016. A motion robust remote-PPG approach to driver's health state monitoring. In *ACCV*, IEEE.

Wu, Qi, Wende Zhang & BVK Vijaya Kumar. 2012. Strong shadow removal via patch-based shadow edge detection. In *ICRA*, IEEE.

Wu, Shangzhe, Christian Rupprecht & Andrea Vedaldi. 2020. Unsupervised learning of probably symmetric deformable 3D objects from images in the wild. In *CVPR*, IEEE.

Wu, Tai-Pang & Chi-Keung Tang. 2005. A bayesian approach for shadow extraction from a single image. In *ICCV*, IEEE.

Wu, Yue, Wael AbdAlmageed & Premkumar Natarajan. 2019. Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features. In *CVPR*, IEEE.

Wu, Yuxin & Kaiming He. 2018. Group normalization. In *ECCV*, .

Xiong, Chao, Xiaowei Zhao, Danhang Tang, Karlekar Jayashree, Shuicheng Yan & Tae-Kyun Kim. 2015. Conditional convolutional neural network for modality-aware face recognition. In *ICCV*, IEEE.

Xiong, Fei & Wael AbdAlmageed. 2018. Unknown presentation attack detection with face rgb images. In *BTAS*, IEEE.

Xu, Zhenqi, Shan Li & Weihong Deng. 2015. Learning temporal features using LSTM-CNN architecture for face anti-spoofing. In *ACPR*, IEEE.

Yang, Jianwei, Zhen Lei & Stan Z Li. 2014. Learn convolutional neural network for face anti-spoofing. *arXiv preprint arXiv:1408.5601* .

Yang, Jianwei, Zhen Lei, Shengcai Liao & Stan Z Li. 2013. Face liveness detection with component dependent descriptor. In *ICB*, IEEE.

Yang, Xiao, Wenhan Luo, Linchao Bao, Yuan Gao, Dihong Gong, Shibao Zheng, Zhifeng Li & Wei Liu. 2019a. Face anti-spoofing: Model matters, so does data. In *CVPR*, IEEE.

Yang, Yang, Xiaojie Guo, Jiayi Ma, Lin Ma & Haibin Ling. 2019b. Lafin: Generative landmark guided face inpainting. *arXiv preprint arXiv:1911.11394* .

Yu, Zitong, Xiaobai Li, Xuesong Niu, Jingang Shi & Guoying Zhao. 2020a. Face anti-spoofing with human material perception. In *ECCV*, .

Yu, Zitong, Chenxu Zhao, Zezheng Wang, Yunxiao Qin, Zhuo Su, Xiaobai Li, Feng Zhou & Guoying Zhao. 2020b. Searching central difference convolutional networks for face anti-spoofing. In *CVPR*, IEEE.

Zhang, Ke-Yue, Taiping Yao, Jian Zhang, Ying Tai, Shouhong Ding, Jilin Li, Feiyue Huang, Haichuan Song & Lizhuang Ma. 2020a. Face anti-spoofing via disentangled representation learning. In *ECCV*, .

Zhang, Li, Tao Xiang & Shaogang Gong. 2017a. Learning a deep embedding model for zero-shot learning. In *CVPR*, IEEE.

Zhang, Shu, Ran He, Zhenan Sun & Tieniu Tan. 2017b. Demeshnet: Blind face inpainting for deep meshface verification. *TIFS* .

Zhang, Xuaner, Jonathan T. Barron, Yun-Ta Tsai, Rohit Pandey, Xiuming Zhang, Ren Ng & David E. Jacobs. 2020b. Portrait shadow manipulation. *TOG* .

Zhang, Zhiwei, Junjie Yan, Sifei Liu, Zhen Lei, Dong Yi & Stan Z Li. 2012. A face antispoofing database with diverse attacks. In *ICB*, IEEE.

Zhang, Ziyuan, Luan Tran, Xi Yin, Yousef Atoum, Jian Wan, Nanxin Wang & Xiaoming Liu. 2019. Gait recognition via disentangled representation learning. In *CVPR*, IEEE.

Zhao, Chenxu, Yunxiao Qin, Zezheng Wang, Tianyu Fu & Hailin Shi. 2019. Meta anti-spoofing: Learning to learn in face anti-spoofing. *arXiv preprint arXiv:1904.12490* .

Zhou, Hao, Sunil Hadap, Kalyan Sunkavalli & David W Jacobs. 2019. Deep single-image portrait relighting. In *ICCV*, IEEE.

Zhu, Xiangyu, Zhen Lei, Xiaoming Liu, Hailin Shi & Stan Z Li. 2016. Face alignment across large poses: A 3D solution. In *CVPR*, IEEE.

Zhu, Xiangyu, Xiaoming Liu, Zhen Lei & Stan Z Li. 2017. Face alignment in full pose range: A 3D total solution. *PAMI* .