Evaluation of a 3D-aided Pose Invariant 2D Face Recognition System

Xiang Xu, Ha A. Le, Pengfei Dou, Yuhang Wu, Ioannis A. Kakadiaris
Computational Biomedicine Lab
4800 Calhoun Rd. Houston, TX, USA
{xxu18, hale4, pdou, ywu35, ikakadia}@central.uh.edu

Abstract

A few well-developed face recognition pipelines have been reported in recent years. Most of the face-related work focuses on a specific module or demonstrates a research idea. In this paper, we present a pose-invariant 3D-aided 2D face recognition system (3D2D-PIFR) that is robust to pose variations as large as 90° by leveraging deep learning technology. We describe the architecture and the interface of 3D2D-PIFR, and introduce each module in detail (Code for module algorithms are kindly provided for the authors and their institution). Experiments are conducted on the UHDB31 and IJB-A, demonstrating that 3D2D-PIFR outperforms existing 2D face recognition systems such as VGG-Face, FaceNet, and a commercial off-the-shelf software (COTS) by at least 9% on UHDB31 and 3% on IJB-A dataset in average. It fills a gap by providing a 3D-aided 2D face recognition system that has compatible results with 2D face recognition systems using deep learning techniques. A video demo of 3D2D-PIFR is available at http://cbl.uh.edu/index.php/pages/research/demos.

1. Introduction

Face recognition is an application in which the computer either classifies human identity according to the face (face identification) or verifies whether two images belong to the same subject (face verification). A common face recognition system has two steps: enrollment and matching. Specifically, in the enrollment stage, features are obtained from a facial image or a set of images to obtain a signature for each subject. The enrollment usually has three steps: (i) detection, (ii) alignment, and (iii) feature representation. In the matching stage, these signatures are compared to obtain a distance for the identification or verification problem. Recently, face recognition technology has significantly advanced due to the deployment of deep learning technology. Pure 2D face recognition (2D-FR) systems have achieved human performance or even better. DeepFace proposed by Taigman *et al*. [33] first reported performance on the Labeled Faces in the Wild (LFW) standard benchmark [14] that was better than human efforts. Schroff *et al*. [31] proposed FaceNet, which used triplet loss to train a deep neural network using 200 million labeled faces, and obtained a performance of 99.63% verification accuracy on the LFW dataset.

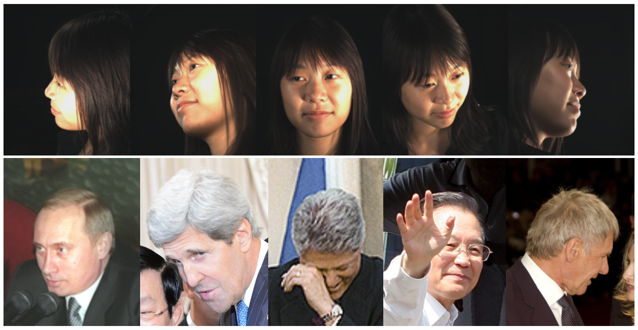

Figure 1: Depiction of existing pose problem from selected samples. Distribution of yaw angles are from $-90°$ to $+90°$ in (T) UHDB31 and (B) IJB-A dataset.

However, face recognition is still not a solved problem in real-world conditions. Some datasets, such as LFW, use Viola-Jones face detector, which is not designed to work in the whole pose distribution from $-90°$ to $+90°$. In an unconstrained scenario, there is a plethora of images with large variations in head pose, expression, illumination, and occlusions. To overcome these challenges, a 3D face model can be applied to assist a 2D face recognition. A 3D facial model is intrinsically invariant to pose and illumination. To use a 3D face model, a model is fit on the facial images and a 3D-2D projection matrix is estimated. With the help of a projection matrix and fitted 3D model, it is easy to align the input images for the feature extraction and matching. In the last few years, a **limited** number of 3D-aided 2D face recognition systems (3D2D-FR) have been developed. Kakadiaris *et al*. [19] proposed a pose and illumination invariant system which frontalized the face image using annotated face model (AFM). Unfortunately, the performance of these systems is far from being compatible with

Name	Category	Core	Detection	Alignment	Representation	Matching	Modern	Active
OpenBR [22]	2D-FR	C++	✓	✓	✓	✓		
FaceID1-3 [32]	2D-FR	-			✓	✓	✓	
DeepFace [33]	2D-FR	-			✓	✓	✓	
FaceNet [31]	2D-FR	-			✓	✓	✓	
VGG-Face [30]	2D-FR	-			✓	✓	✓	
OpenFace [5]	2D-FR	Torch	✓	✓	✓	✓	✓	
U-3DMM [12]	3D2D-FR	-			✓	✓	✓	
3D2D-PIFR	3D2D-FR	C++	✓	✓	✓	✓	✓	✓

Table 1: Comparison of recent existing 2D face recognition pipelines. We employ the same definition of *modern* and *active* made by Klontz *et al*. [22]. "-" means that this information is not provided in the paper.

2D-FR using deep learning. Hu *et al*. [12] proposed a unified 3D morphable model (U-3DMM) which has additional PCA subspace for perturbation.

To address the problem mentioned above, in this paper we present a 3D-aided 2D face recognition system called 3D2D-PIFR, which significantly improves face recognition performance using 3D face model and deep learning technology, especially in large pose scenarios. There is enormous demand for pose-invariant face recognition systems because frontal face recognition is a solved problem. 3D2D-PIFR consists of several independent modules: face detection, landmark detection, 3D model reconstruction, pose estimation, lifting texture, feature representation, and matching. It provides sufficient tools and interfaces to use different sub-modules designed in the system. The core code is written in efficient C++, which provides bindings to Python. The system leverages several open-sourced libraries such as OpenCV [3], glog [2], gflags [1], pugixml [4], JSON for modern C++ and Caffe [16]. In 3D2D-PIFR, a 3D model is constructed from a 2D image. By estimating the 3D-2D projection matrix, the correspondence between the 3D model and 2D image can be computed. Then, a 3D model is used to help to frontalize the face. The pose robust features and occlusion encodings are extracted to represent the face. For matching, we use cosine similarity to compute the similarity between two feature vectors.

In summary, in this paper we make the following contributions:

- We developed a well-designed, pose-invariant, 3D-aided 2D face recognition system using deep learning. The intrinsic value of a 3D model is explored to frontalize the face, and the pose-invariant features are extracted for representation.

- We demonstrate that a 3D-aided 2D face recognition system exhibits a performance that is comparable to a 2D only FR system. Our face recognition results outperform the VGG-Face, FaceNet, and COTS by at least 9% on UHDB31 and 3% on IJB-A dataset in average.

The rest of the paper is organized as follows: modern face recognition systems are reviewed in Sec. 2. In Sec. 3, we discuss the architecture of 3D2D-PIFR and its functionalities. In Sec. 4, we introduce each module separately in detail. Detailed evaluations on the indoor and in-the-wild datasets are reported in Sec. 5.

2. Related work

We divide the current existing face-related work into two categories: In Sec. 2.1, we discuss recent work for each module in the common face recognition pipeline. Then, the system level papers are discussed in Sec. 2.2.

2.1. Modules

Detection: Face detection is the first step as well as the most studied topic in the face recognition domain. Zefeiriou *et al*. [40] presented a comprehensive survey on this topic. They divided the approaches into two categories: rigid template-based methods, and deformable parts models. Despite the methods summarized in [40], the approaches of object detection under the regions with convolutional neural network (R-CNN) framework [9] have been well developed. Some techniques can be directly integrated to face detection [17]. Li *et al*. [26] used a 3D mean face model and divided the face into ten parts. They joined face proposals into a single R-CNN model. The approach proposed by Hu and Ramanan [13] explored context and resolution of images to fine-tune the residual networks (ResNet) [11], which was demonstrated to detect a face as small as 3 pixels.

Alignment: Face alignment refers to aligning the face image to a specific position. Usually, researchers include landmark detection in this topic. Jin and Tan [18] summarized the categories of popular approaches for this task. Zhu *et al*. [42] searched for similar shapes from exemplars and regressed the shapes by using SIFT features and updating the probability of shapes. Xu and Kakadiaris [37] proposed to jointly learn head pose estimation and face alignment tasks in a single framework (JFA) using global and local CNN features. KEPLER [24] joined CNN features from different layers and captured the response map to localize the landmarks. Wu *et al*. [36] proposed GoDP algorithm to localize landmarks under a fully convolution network (FCN) framework by exploring two-pathway information. Some recent works use generative adversarial networks (GAN) to frontalize the face [15, 39]. Huang *et al*. [15] used two-pathway GAN (TP-GAN) for photo-realistic frontal synthesis images, but kept identity and details. Yin *et al*. [39] incorporated a 3D model with GAN to frontalize faces for large poses in the wild.

Representation: An emerging topic in face recognition research is finding a discriminative representation for a subject. When training with millions of face images using deep learning technology, many feature descriptors have been proposed recently. Parkhi *et al*. [30] proposed the VGG-Face descriptor within VGG-Very-Deep architectures. Schroff *et al*. [31] proposed triplet loss to train a deep neural network using 200 million labeled faces from Google. Hu *et al*. [12] proposed Unified 3D morphable model (U-3DMM), which has additional PCA subspace for perturbation. Masi *et al*. [27] developed face recognition for unconstrained environments by fine-tuning the ResNet and VGG-Face on 500K 3D rendering images. In addition to frontalizing the face, they also rendered face images to half-profile $40°$, and full-profile $(75°)$. Masi *et al*. [28] addressed the question of whether we need to collect millions of faces for training a face recognition system. They argued that we can use synthesized images instead of real images to train the model and still obtain the comparable results. Wen *et al*. [34] added center loss alongside cross entropy loss to obtain discriminative features for deep face recognition.

2.2. System

OpenCV and OpenBR are some well known open-sourced computer vision and pattern recognition libraries. However, the eigenface algorithm in the OpenCV is out-of-date. OpenBR is no longer updated since 9/29/2015. Both of them only support nearly frontal face recognition. OpenFace is an open-sourced implementation of FaceNet [31] by Amos *et al*. [5] using Python and Torch, which provides four demos for usage.

To the best of our knowledge, there is a limited amount of well-designed system papers. Most of the face-related papers focus on the different sub-modules or research of face recognition. The comparison of recent existing 2D face recognition system is presented in Tab. 1 including the research on face representation.

3. System Design

3D2D-PIFR is a 3D-aided 2D face recognition system designed for pose-invariant face recognition. Moreover, this system is suitable for face-related research, and can fast preprocess images, provide baselines, plot the results, and support further development.

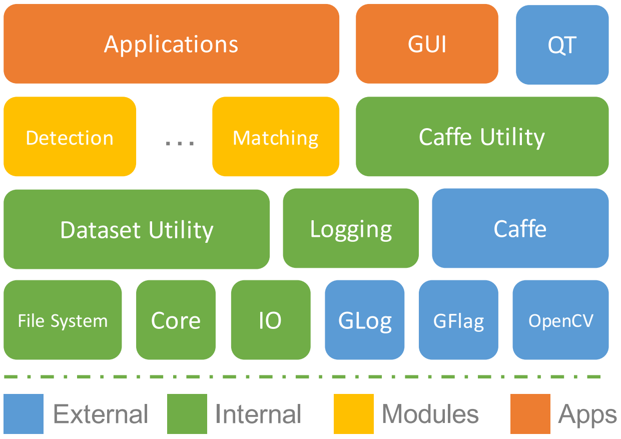

Figure 2: Depiction of 3D2D-PIFR's architecture. In addition to external libraries, it includes some other base libraries to process files, use CUDA, manage the data files, *etc*. Based on these basic libraries, high level APIs were implemented by calling function from each module. Based on 3D2D-PIFR's SDK, it is easier to write various applications for different purposes. Also, we created the GUIs to demonstrate our 3D2D-PIFR.

3.1. Pre-requirements

3D2D-PIFR is written in clean and efficient C++, which is developed in Linux platform (Ubuntu system). It requires GCC 4.9 or above for compilation. It leverages a list of open-sourced libraries and tools such as CMake, Boost, OpenCV, gflags, glog, puxixml, JSON, and Caffe. Most of the dependencies are available in the Ubuntu repository except Caffe. Therefore, to install dependencies, it only requires installing Caffe manually.

3.2. Architecture Overview

Figure 2 illustrates the architecture of 3D2D-PIFR, which explicitly illustrates modules and functionality. The blue blocks are external shared libraries. The other three components belong to our system. As a base of the software, green blocks provide the basic functions. The algorithm modules are constructed as high-level APIs. The applications and GUIs are the top of the software and are built by combining these APIs. The users can directly call these applications and obtain the results. The advantages of this

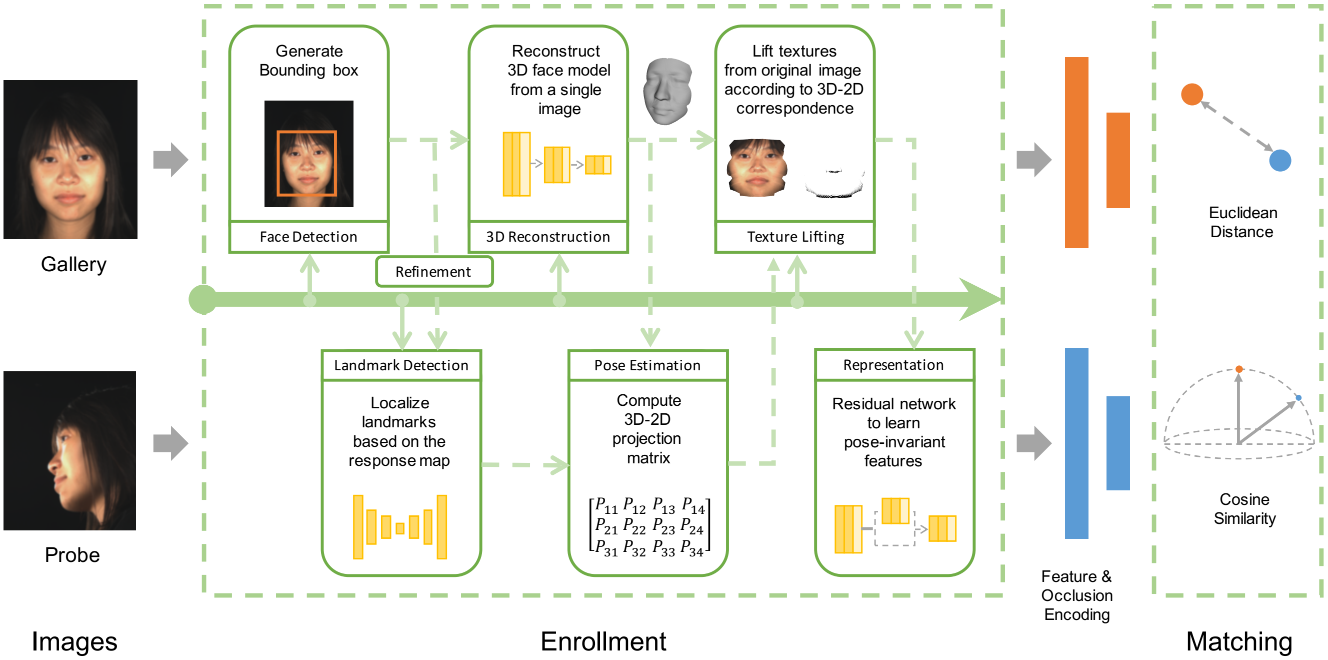
Gallery
Probe
Images
Generate Bounding box
Face Detection
Reconstruct 3D face model from a single image
3D Reconstruction
Lift textures from original image according to 3D-2D correspondence
Texture Lifting
Refinement
Landmark Detection
Localize landmarks based on the response map
Pose Estimation
Compute 3D-2D projection matrix
$\begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix}$
Representation
Residual network to learn pose-invariant features
Enrollment
Feature & Occlusion Encoding
Euclidean Distance
Cosine Similarity
Matching

Figure 3: Depiction of the whole pipeline (follow the arrow in the middle) of 3D2D-PIFR. The rounded rectangles represent different modules. Dashed arrows represent the workflow. The enrollment encompasses the modules listed. A face first is detected then is transferred to localize landmarks. A 3D model is constructed directly from a 2D image with a bounding box. With 2D landmarks and a 3D model, a 3D-2D projection matrix can be estimated. The frontalized image and occlusion map are generated according to the 3D model and projection matrix. The pose robust features are extracted from these images along with occlusion encoding. The matching step computes features from visible parts and outputs a similarity score.

architecture are simplicity and well structured. With full development of libraries, the system can use CPUs/GPU and other features easily.

3.3. Data Structures

In 3D2D-PIFR, the basic element is `File` on the disk. All operations or algorithms are based on the files. The basic data structure is `Data`, which is a hash table with pairs of keys and values. Both keys and values are in string type. Unlike OpenBR [22], to avoid saving giant data in the memory, we only keep the file path in the memory.

3.4. Configuration

We have two approaches to run 3D2D-PIFR. The first one is defining the configuration file (JSON format), which points out the datasets, input files, output directories, involving modules and their model locations, and evaluation. Attribute `dataset` contains the information of input dataset including the name and path. Attribute `input` contains the list of galleries and probes. Attribute `output` defines the output directories. Attribute `pipelines` defines the modules used in the pipeline. The `pip` command line application only accepts the argument of the configuration file, which will parse the configuration file, load the models, and run defined modules. The advantages of this approach are simplicity and flexibility. Unlike the OpenBR framework, it does not require a detailed understanding of the option or input long arguments in the command line. The users only need to change some values in the attributes `dataset` and `input` (*e.g.*, set dataset directory and file to enroll), and program `pip` will generate the output they defined in this configuration file.

3.5. Command Line Interface

To make full use of SDK of 3D2D-PIFR, we created some corresponding applications to run each module. All applications accept the file list (text or csv file by default, which includes tag at the top line), a folder, or a single image. The `IO` system will load the data in the memory and process the data according to the data list.

The arguments specify the location of the input file/directory and where the output should be saved. The 3D2D-PIFR's enrollment is executed and generates signatures to the output directory. The path of the signature is recorded in the `Data`. By calling the API from `IO` system, the list of `Data` will be written to the file (default is in

.csv format).

4. Face Recognition

4.1. Face Detection

A serious problem in OpenBR [22], OpenFace [5], and even the commercial off-the-shelf face recognition software (COTS) is the face detection rate. OpenBR only supports OpenCV frontal face detector. OpenFace also supports Dlib [20] face detector. However, In recent years, a lot of face detection algorithms has been developed [8, 26, 13] with deep learning technology to support multi-view images.

To detect the face in multi-view poses, some modern detectors such as Headhunter [29] and DDFD [8] are supported in our system. Mathias *et al*. [29] trained Headhunter by using multi-scale templates. DDFD face detector is proposed by Farfade *et al*. [8] by fine-tuning AlexNet [23] and using non-maximum suppression (NMS-max, NMS-avg). In addition, we wrap the Dlib-DNN face detector in 3D2D-PIFR for 2D face detection.

To support different face detectors for downstream modules, we perform the bounding box regression on detected bounding box to reduce the variations of the bounding box. The first advantage of this approach is we do not need to retrain or fine-tune the models for downstream modules after switching the face detector. The second advantage is that it provides a more robust bounding box for the landmark localization module.

4.2. Landmark Localization

To detect the face landmarks, we use GoDP proposed by Wu *et al*. [36], which is demonstrated to be robust to pose variations. GoDP landmark detector relies on confidence maps generated by a fully convolutional network. A confidence map is generated for each landmark to indicate the possibility of a landmark appearing at a specific location in the original image. The prediction is made by simply selecting the location that has the maximum response in the confidence map. This winner-take-all strategy helps to suppress false alarms generated by background regions and improves the robustness of the algorithm under large head pose variations. Compared to other confidence-map-based landmark detectors, the novel architecture of GoDP merges the information of the deep and shallow layers based on a new loss function, increases the resolution and discrimination of the confidence maps, and achieves state-of-the-art results on multiple challenging face alignment databases.

4.3. 3D Reconstruction

To reconstruct the 3D facial shape of the input 2D image, we integrate into our pipeline the E2FAR algorithm proposed by Dou *et al*. [6]. It uses a subspace model to represent a 3D AFM as a parameter vector and employs CNN to estimate the optimal parameter values from a single 2D image. To train the deep neural network, a large set of synthetic 2D and 3D data has been created using the 3D rendering of randomly generated AFMs. To improve the robustness to illumination variation, the deep neural network is pre-trained on real facial images and fine-tuned on the synthetic data. Compared with existing work, it is more efficient due to its end-to-end architecture, which requires a single feed-forward operation to predict the model parameters. Moreover, it only relies on face detection to localize the facial region of interest on the image. As a result, compared with landmark-based approaches, it is more robust to the pose variation that can degrade landmark detection accuracy.

4.4. Pose estimation

Given 2D landmarks X_{2D} obtained from landmark detection and 3D landmarks X_{3D} obtained from 3D model, the transformation matrix P can be estimated by solving a least-squares problem as follows:

$$\min_{P} ||X_{2D} - PX_{3D}||_2^2. \qquad (1)$$

4.5. Texture Lifting

Facial texture lifting is a technique first proposed by Kakadiaris *et al*. [19], which lifts the pixel values from the original 2D images to a UV map. Given the 3D-2D projection matrix P, 3D AFM model M, and original image I, it first generates the geometry image G, each pixel of which captures the information of an existing or interpolated vertex on the 3D AFM surface. With G, a set of 2D coordinates referring to the pixels on an original 2D facial image is computed. In this way, the facial appearance is lifted and represented into a new texture image T. A 3D model M and Z-Buffer technique are used to estimate the occlusion status for each pixel. This process generates an occlusion mask Z.

This module has the following two advantages: It generates the frontal normalized face images, which is convenient for feature extraction and comparison. Second, it generates occlusion masks, which identify the parts of the face images that are occluded, providing give evidence to exclude the face regions.

4.6. Representation

To improve the performance of face recognition in matching non-frontal facial images, we integrate into our pipeline the algorithm proposed by Dou et al. [7] for extracting Pose-Robust Face Signature (PRFS), a part-based face representation with discriminative local facial features and explicit pose and self-occlusion encoding. The facial texture T and the self-occlusion mask Z are divided into multiple local patches first. Then, on each local patch,

Dataset	Images	Subjects	Environment	Poses	Illuminations	Usage
UHDB31	24,255	77	Constrained	21	3	2D-2D, 3D-2D, 3D-3D face recognition
IJB-A	25,808	500	In-the-wild	Various	Various	2D unconstrained face recognition

Table 2: Comparison of two datasets. Both are challenging due to pose variations, illumination, and resolution.

discriminative features are extracted and self-occlusion encoding is computed. The ensemble of local features, each enhanced by the self-occlusion encoding, forms the pose-robust face signature. We use two types of local feature, namely the DFD feature proposed by Lei et al. [25] and a deep feature we trained by following [34]. To train the DFD feature, we use a small subset of the FRGC2 database that consists of 907 frontal facial images of 109 subjects. We divide the facial texture into 64 non-overlapping patches and train a DFD feature extractor for each local patch separately. To train the deep feature, we use the CASIA WebFace dataset [38] as training data. We divide the facial texture into 8 partially-overlapping patches and train a deep neural network for each local patch separately. In this paper, we call the face signature with the DFD feature PRFS, and the face signature with the deep feature DPRFS.

5. Experiments

In this section, we provide a system and numerical analysis on two challenging datasets in both constrained and in-the-wild scenarios. First, we introduce the datasets used to verify 3D2D-PIFR. Then, we compare 3D2D-PIFRwith VGG face descriptor (VGG-Face) and a commercial face recognition software (COTS) on these two challenging datasets.

5.1. Datasets

UHDB31 [35] is created in the controlled lab environment, which allows face-related research on pose and illumination issues. In addition to 2D images, it also provides the corresponding 3D model of subjects. An interesting fact of this dataset is that pose follows the uniform distribution on 3 dimensions: pitch, yaw, and roll. For each subject, a total of 21 high-resolution 2D images from different views and 3D data are collected at the same time. Then, a 3D model is registered from the 3D data from different poses to generate a specific 3D face model. In addition to three illuminations, the resolutions are downsampled to 128, 256, and 512 from the original size.

IJB-A [21] is another challenging dataset which consists of images in the wild. This dataset was proposed by IARPA and is managed by NIST. This dataset merges images and frames together and provides evaluations on the template level. A template contains one or several images/frames of a subject. According to the IJB-A protocol, it splits galleries and probes into 10 folders. In our experiment, we modify this protocol to use it for close-set face identification. The details will be introduced in Sec. 5.4. A summary of these two datasets is presented in Tab. 2. Our system provides `dataset utility` to parse and load the data from these two datasets.

System	Features	Dims	Metric
VGG-Face	Embedding	4096	Cosine
COTS v1.9	-	-	-
FaceNet	-	128	Cosine
3D2D-PIFR	PRFS	64×1024	Cosine
3D2D-PIFR	DPRFS	8×1024	Cosine

Table 3: Comparison of systems configuration used in our experiments.

5.2. Baselines

To perform a fair comparison with current state-of-the-art face recognition systems, we choose VGG-Face and COTS v1.9 as baselines.

The VGG-Face descriptor was developed by Parkhi *et al.* [30]. The original release contains a Caffe model and a MATLAB example. We re-used their model and implemented their embedding method on multi-scaled images and fused the features in C++. In our implementation, we tried different combinations of descriptor and matching methods. We found that embedding features with cosine similarity metric work the best for the VGG-Face. In our experiment, we use VGG-Face to represent the embedding features with matching using cosine similarity metric. As in the baseline module, 3D2D-PIFR provides API to obtain the features.

The FaceNet algorithm was proposed by Schroff *et al.* [31]. We use a personal implemented FaceNet from GitHub [1]. They first use MTCNN [41] to align face and extract features using FaceNet 128 dimensions features. They provide the pre-trained model that achieves 99.3%±0.004% accuracy on LFW dataset trained using WebFace [38] and MS-Celeb-1M [10]. The accuracy is a little bit lower than the original paper, but still, can be thought as state-of-the-art.

[1] https://github.com/davidsandberg/facenet

Yaw Pitch	$-90°$	$-60°$	$-30°$	$0°$	$+30°$	$+60°$	$+90°$
$+30°$	14/11/58/47/**82**	69/32/95/90/**99**	94/90/**100**/**100**/**100**	99/**100**/**100**/**100**/**100**	95/93/99/**100**/99	79/38/92/95/**99**	19/7/60/47/**75**
$0°$	22/9/84/81/**96**	88/52/99/**100**/**100**	**100**/99/**100**/**100**/**100**	-	**100**/**100**/**100**/**100**/**100**	94/73/99/100/**100**	27/10/91/84/**96**
$-30°$	8/0/44/44/**74**	2/19/80/90/**97**	91/90/99/99/**100**	96/99/99/**100**/**100**	96/98/97/99/**100**	52/15/90/95/**96**	9/3/35/58/**78**

Table 4: Comparison of Rank-1 of different systems on UHDB31.R128.I03. The methods are ordered as VGG-Face, COTS v1.9, FaceNet, 3D2D-PIFR-PRFS, and 3D2D-PIFR-DPRFS. We drop $\%$ for simplicity. The index of poses are ordered from the left to right and from the top to bottom (*e.g.*, pose 3 is pitch $-30°$ and yaw $-90°$, pose 11 is pitch $0°$ and yaw $0°$). The frontal face is gallery while the other poses are probes. In all cases, our system achieves the best performance compared with the state-of-the-art.

COTS is a commercial software developed for scalable face recognition. It provides SDK and applications which can be directly used. In our experiments, we used version 1.9 to compare with our system. This version is considered a significant boost compared with previous versions.

In our experiment, we report the performance using both PRFS and DPRFS features. The summary of software configuration is reported in Tab. 3. We compute the Rank-1 identity accuracy from successfully enrolled signatures.

5.3. UHDB31: Pose-Invariant face Recognition

In this experiment, we chose a configuration from UHDB31 named UHDB31.R0128.I03. This is a subset in which all images are down-sampled to the size 153×128 in the neutral illumination. This subset was chosen to demonstrate that our system, 3D2D-PIFR, is robust to different poses. Therefore, we use this configuration to exclude the other variations such as illumination, expressions, *etc*., but only keep the pose variations.

We treated the frontal face images (pose 11) as gallery and images from the other 20 poses (poses $1-10$, $12-21$) as probes, independently. Both the gallery and the probe contain 77 images, each of which belongs to a subject. The face identification experiment was performed using 20 pairs of sigsets.

Table 4 depicts the comparison of Rank-1 accuracy among 20 poses (except pose 11, which is used for gallery), which indicates that 3D2D-PIFR is robust to the different poses compared with other systems. We observed the VGG-Face and COTS v1.9 algorithms cannot generalize all pose distributions. FaceNet works better than VGG-Face and COTS v1.9 on the extreme poses. One possible answer is that this model is trained from the most available datasets using Ms-Celeb-1M and WebFace, which provide more extreme pose cases. However in cases such as pose 3 $(-30°, -90°)$ and pose 21 $(-30°, -90°)$ in Tab. 4, the performance of 2D only face recognition pipelines still have large room to improve. On the other hand, with the help of the 3D model, our system keeps the consistent and symmetric performance among the different poses. Even in the cases with yaw $-90°$ or $+90°$, our system can tolerate the pose variations, and achieves around 80% Rank-1 identity accuracy with DPRFS features and around 50% Rank-1 identity accuracy with PRFS features in average.

5.4. IJB-A: In-the-wild Face Recognition

However, in a real-world case, a face recognition system does not suffer only from pose variations. In this experiment, we want to explore whether our system is can also be used in an in-the-wild environment. We designed a different protocol for face identification experiments based on the original 10 splits. Unlike the original template-level comparison, we conducted an image pairs comparison. First, we removed some samples in the IJB-A splits to make 10 close-set comparison pairs. Then, we cropped the face according to the annotations. Image thumbnails with resolution below 50 were up-sampled, while those with resolution larger than 1000 were down-sampled. Herein, we do not compare with FaceNet since there is overlapping samples between the training set and IJB-A dataset.

Table 5 depicts the rank-1 identification rate with different methods on IJB-A dataset. Our system 3D2D-PIFR with DPRFS reports better performance compared with VGG-Face and COTS v1.9. Also, our system results are consistent on 10 splits, which indicates that our system is robust. Why do PRFS features in our system not perform well on the IJB-A dataset? One possible answer is that PRFS features are trained on the FRGC dataset, which has much fewer variations of pose, illumination, and resolution problems. The current PRFS features cannot generalize on these images with large variances. The corresponding solution is retraining the PRFS feature model on the in-the-wild dataset. Third, COTS performs well on this challenging datasets, since it is designed for the real scenario. Finally,

Method	Split-1	Split-2	Split-3	Split-4	Split-5	Split-6	Split-7	Split-8	Split-9	Split-10	Avg.
VGG-Face	76.18	74.37	24.33	47.67	52.07	47.11	58.31	54.31	47.98	49.06	53.16
COTS v1.9	75.68	76.57	73.66	76.73	76.31	77.21	76.27	74.50	72.52	77.88	75.73
Ours-PRFS	47.61	49.27	47.71	47.71	48.97	44.83	52.98	44.14	43.40	49.02	47.56
Ours-DPRFS	**78.20**	**76.97**	**77.31**	**79.00**	**78.01**	**79.00**	**81.15**	**78.40**	**74.97**	**78.57**	**78.16**

Table 5: Comparison of Rank 1 of different systems on 10 splits of IJB-A. 3D2D-PIFR achieves the best performance with DPRFS features on each split. We drop % for simplicity.

we come out a question that comparing the experiment in Sec. 5.3, why our system only outperforms slightly better than baselines? We argue that in the wild scenarios, there are complicated combinations of pose variations, illumination, expression, and occlusions. A robust face recognition system should take all cases into consideration. In addition, COTS dropped hard samples and enrolled less signatures than ours, which would boost the performance to some extent.

5.5. Memory Overhead and Running Time

We conducted the analysis of 3D2D-PIFR in terms of both memory and time. Caffe-related implementation runs on GPU (GTX TITAN X). COTS v1.9 makes full use of 8 CPUs. Table 6 summarizes the system run-times for different systems. Some modules of our implementation or external libraries run on CPU, such as face detection, pose estimation, text-lifting, and PRFS feature extraction. Therefore, the time using PRFS features takes 1.5 s more than using DPRFS features. Due to loading several large models, DPRFS requires more memory. The user can define the suitable feature extractors according to their needs.

System	GPU	Memory (GB)	Time (s)
VGG-Face	Full	1.2	0.9
COTS v1.9	No	0.1	0.5
3D2D-PIFR-PRFS	Half	2.4	2.5
3D2D-PIFR-DPRFS	Half	5.0	1.0

Table 6: Comparison of system run-times. "Half" in GPU column means part of the code does not support GPU acceleration. Time means the average enrollment time for a single image.

6. Conclusion

In this paper, we present a well-designed 3D-aided 2D face recognition system (3D2D-PIFR) that is robust to pose variations as large as 90° using deep learning technology. We overviewed the architecture and interface of 3D2D-PIFR and introduce each module in the pipeline. Detailed experiments were conducted on UHDB31 and IJB-A to demonstrate that 3D2D-PIFR is robust to the pose variations, and it outperforms over the existing 2D only face recognition systems such as VGG face descriptor, FaceNet, and a commercial face recognition software.

7. Acknowledgment

This material is based upon work supported by the U.S. Department of Homeland Security under Grant Award Number 2015-ST-061-BSH001. This grant is awarded to the Borders, Trade, and Immigration (BTI) Institute: A DHS Center of Excellence led by the University of Houston, and includes support for the project "Image and Video Person Identification in an Operational Environment" awarded to the University of Houston. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security.

References

[1] *Gflags*, https://github.com/gflags/gflags. 2

[2] *Glog*, https://github.com/google/glog. 2

[3] *OpenCV*, http://opencv.org. 2

[4] *Pugixml*, https://github.com/zeux/pugixml. 2

[5] B. Amos, B. Ludwiczuk, and S. Mahadev. OpenFace: A general-purpose face recognition library with mobile applications. Technical Report CMU-CS-16-118, CMU School of Computer Science, Pittsburgh, PA, 2016. 2, 3, 5

[6] P. Dou, S. K. Shah, and I. A. Kakadiaris. End-to-end 3D face reconstruction with deep neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Honolulu, Hawaii, July 22-25 2017. 5

[7] P. Dou, L. Zhang, Y. Wu, S. K. Shah, and I. A. Kakadiaris. Pose-robust face signature for multi-view face recognition. In *Proc. International Conference on Biometrics: Theory, Applications and Systems*, pages 1–8, Arlington, VA, Sep. 8-11 2015. 5

[8] S. S. Farfade, M. Saberian, and L. Li. Multi-view face detection using deep convolutional neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Boston, Massachusetts, June 7 - 12 2015. 5

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Columbus, OH, June 24-27 2014. 2

[10] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao. MS-Celeb-1M: A dataset and benchmark for large-scale face recognition. In *Proc.* 14^{th} *European Conference on Computer Vision*, Amsterdam, Netherlands, Oct. 11-16 2016. 6

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, Las Vegas, NV, June 26-July 1 2016. 2

[12] G. Hu, F. Yan, C. Chan, W. Deng, W. Christmas, J. Kittler, and N. M. Robertson. Face recognition using a unified 3D morphable model. In *Proc.* 14^{th} *European Conference on Computer Vision*, Amsterdam, Netherlands, Oct. 11-16 2016. 2, 3

[13] P. Hu and D. Ramanan. Finding tiny faces. *ArXiv e-prints*, pages 1–13, 2016. 2, 5

[14] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller. Labeled faces in the Wild: A database for studying face recognition in unconstrained environments. In *Proc. European Conference on Computer Vision*, Marseille, France, Oct. 17-20 2008. 1

[15] R. Huang, S. Zhang, T. Li, and R. He. Beyond face rotation: global and local perception gan for photorealistic and identity preserving frontal view synthesis. *ArXiv eprint*, pages 1–11, 2017. 3

[16] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. B. Girshick, S. Guadarrama, and T. Darrell. Caffe: convolutional architecture for fast feature embedding. In *Proc.* 22^{nd} *international conference on Multimedia*, pages 675–678, Orlando, Florida, USA, Nov. 03 - 07 2014. 2

[17] H. Jiang and E. Learned-Miller. Face detection with the faster r-cnn. *ArXiv e-prints*, pages 1–6, 2016. 2

[18] X. Jin and X. Tan. Face alignment in-the-wild: A survey. *ArXiv eprint*, pages 1–38, Aug. 15 2016. 2

[19] I. A. Kakadiaris, G. Toderici, G. Evangelopoulos, G. Passalis, D. Chu, X. Zhao, S. K. Shah, and T. Theoharis. 3D-2D face recognition with pose-illumination normalization. *Computer Visiona and Image Understanding*, 154:137–151, 2017. 1, 5

[20] D. E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009. 5

[21] B. F. Klare, B. Klein, E. Taborsky, A. Blanton, J. Cheney, K. Allen, P. Grother, A. Mah, M. Burge, and A. K. Jain. Pushing the frontiers of unconstrained face detection and recognition: IARPA janus benchmark A. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1931–1939, Boston, Massachusetts, June 7 - 12 2015. 6

[22] J. Klontz, B. Klare, S. Klum, A. Jain, and M. Burge. Open source biometric recognition. In *Proc. IEEE Conference on Biometrics: Theory, Applications and Systems*, Washington DC, September 29 - October 2 2013. 2, 4, 5

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Proc. Advances in neural information processing systems*, pages 1097–1105, 15 U.S. 50, Stateline, NV 89449, 12 2012. 5

[24] A. Kumar, A. Alavi, and R. Chellappa. KEPLER: keypoint and pose estimation of unconstrained faces by learning efficient h-cnn regressors. In *Proc.* 12^{th} *IEEE Conference on Automatic Face and Gesture Recognition*, Washington, DC, May 30 June 3 2017. 3

[25] Z. Lei, M. Pietikainen, and S. Li. Learning discriminant face descriptor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(2):289–302, Feb. 2014. 6

[26] Y. Li, B. Sun, T. Wu, and Y. Wang. Face detection with end-to-end integration of a ConvNet and a 3D model. In *Proc.* 14^{th} *European Conference on Computer Vision*, Amsterdam, Netherlands, Oct. 11-16 2016. 2, 5

[27] I. Masi, S. Rawls, G. Medioni, and P. Natarajan. Pose-aware face recognition in the wild. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Las Vegas, NV, June 26-July 1 2016. 3

[28] I. Masi, A. Trn, T. Hassner, J. Leksut, and G. Medioni. Do we really need to collect millions of faces for effective face recognition? In *Proc. European Conference on Computer Vision*, pages 579–596, Amsterdam, The Netherlands, October 11-14 2016. 3

[29] M. Mathias, R. Benenson, M. Pedersoli, and L. V. Gool. Face detection without bells and whistles. In *Proc. 13th European Conference on Computer Vision*, pages 720–735, Zurich, Switzerland, Sep. 6-12 2014. 5

[30] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. In *Proc. British Machine Vision Conference*, pages 1–12, Swansea, UK, September 7-10 2015. 2, 3, 6

[31] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A unified embedding for face recognition and clustering. *Proc. Computer Vision and Pattern Recognition*, pages 815–823, jun 2015. 1, 2, 3, 6

[32] Y. Sun, D. Liang, X. Wang, and X. Tang. DeepID3: face recognition with very deep neural networks. *arXiv preprint arXiv:1502.00873*, pages 1–5, 2015. 2

[33] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. DeepFace: Closing the gap to human-level performance in face verification. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 1701 – 1708, Columbus, Ohio, June 24-27 2014. 1, 2

[34] Y. Wen, K. Zhang, Z. Li, and Y. Qiao. A discriminative feature learning approach for deep face recognition. In *Proc.* 14^{th} *European Conference on Computer Vision*, pages 499–515, Amsterdam, Netherlands, Oct. 11-16 2016. 3, 6

[35] Y. Wu, S. K. Shah, and I. A. Kakadiaris. Rendering or normalization? An analysis of the 3D-aided pose-invariant face recognition. In *Proc. IEEE International Conference on Identity, Security and Behavior Analysis*, pages 1–8, Sendai, Japan, February 29-March 2 2016. 6

[36] Y. Wu, S. K. Shah, and I. A. Kakadiaris. GoDP: globally optimized dual pathway system for facial landmark localization in-the-wild. *ArXiv eprint*, pages 1–16, 2017. 3, 5

[37] X. Xu and I. A. Kakadiaris. Joint head pose estimation and face alignment framework using global and local CNN features. In *Proc.* 12^{th} *IEEE Conference on Automatic Face*

and Gesture Recognition, Washington, DC, May 30-June 3 2017. 2

[38] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Learning face representation from scratch. *ArXiv e-prints*, pages 1–9, November 2014. 6

[39] X. Yin, X. Yu, K. Sohn, X. Liu, and M. Chandraker. Towards large-pose face frontalization in the wild. *ArXiv eprint*, pages 1–12, 2017. 3

[40] S. Zafeiriou, C. Zhang, and Z. Zhang. A survey on face detection in the wild: past, present and future. *Computer Vision and Image Understanding*, 138:1–24, Sept. 2015. 2

[41] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multitask cascaded convolutional networks. *IEEE Signal Processing Letters*, 23(10):1499–1503, 2016. 6

[42] S. Zhu, C. Li, C. C. Loy, and X. Tang. Face alignment by coarse-to-fine shape searching. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, pages 4998–5006, Boston, MA, June 7 - 12 2015. 2