Face Detection and Recognition for Fish-Tank Virtual Reality by Deep Network

Kaseya Xia
University of British Columbia
Department of Electrical and Computer Engineering
Vancouver, Canada
Zxia0101@student.ubc.ca

***Abstract*— 3D teleconferencing is one of the most fast-emerging technologies that improve the overall conference meeting experience for users. Virtual Reality and Augmented Reality are the two main approaches researchers have been focusing on to support the development of 3D teleconferencing. University of British Columbia Human Computer Interaction lab has recently developed Fish Tank Virtual Reality with affordable materials and provides Virtual Reality experience to two users with headset wearing. But in order to apply this system to 3D teleconferencing, many challenges like latency, accurate user face tracking, and multiple user interfaces have to be tackled. In this paper, we present a face detection and recognition method based on deep network and RGB-D input data with the goal of fast detection rate, invariance to illumination, and robustness in detection distance. We propose a face detection network based on the Multi-task Convolutional Neural Network which leverages the depth map detection to increase detection robustness. For face recognition, we perform transfer learning on 3 user input videos on VGG-16 network to mimic the FTVR user on site experience. Our system showed robustness to detection distance and invariance to illumination with RGB input. However, the depth detection network and the detection speed still need improvement. The face recognition demonstrates 80% accurate rate for single user but higher false negative rate for two user cases.**

Keywords—3D-teleconferencing, Virtual Reality, face detection, face recognition, latency, depth camera, deep learning, transfer learning.

I. Introduction

Under the impact of COVID-19 pandemic, there has been a resurgence in need of new teleconferencing technology that creates a more real and in-person experience. A 3D teleconferencing that allows the user to appear in 3D and maintain direct eye contact with multiple speakers will enhance the overall communication experience and improve the information transmission efficiency [3]. Virtual Reality (VR) and Augmented Reality (AR) are the two main trends to create 3D effects in recent years. Three different types of hardware have also emerged to support the creation of those two realities: headsets that connect to your PC, 2D semi-transparent displays like Google Glasses, and standalone 3D display devices. In the last five years, tremendous amounts of inventors and researchers have flowed into this field; after Mark Zuckerburg bought Oculus for two billion dollars, Oculus VR headsets have becoming one of the most popular and affordable commodities on the market [4]. Unlike headset VR, FTVR creates a 3D illusion environment that allows the user to easily transit between real world information and illusional world information, which is more suitable for 3D teleconferencing. Mobile phone AR creates similar virtual information via 2D display, but due to the nature of overlaying display on real objects AR effects are not as tangible VR effects [1]. University of British Columbia (UBC) Human Computer Interaction Lab (HCI) has built a Fish Tank Virtual Reality (FTVR) system with affordable materials and it currently supports two user interfaces at the same time. UBC FTVR uses projection from different pico-projectors under a crystal ball display to create the VR effect. Our FTVR leverages semi-automatic approaches and pre-calibration technique to achieve an automatic calibration system with less than 1mm on-surface point error [2]. However, in order to minimize the distortion effect introduced by the latency between the user movement and projection calibration system, users need to wear physical headset to provide head positions to the calibration system. To further push FTVR to deployment in 3D teleconferencing application, this paper investigates a face detection and

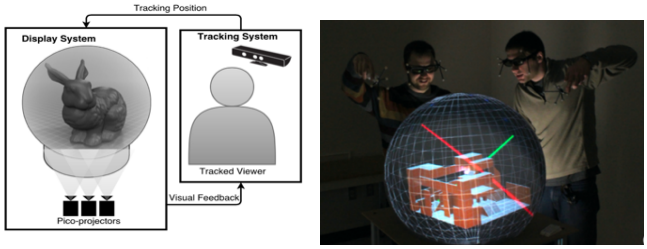

Fig. 1. (Left) UBC FTVR system that supports two different user interfaces with headsets. (Right) Proposed face tracking and feedback calibration system: Rather than equipping headsets, GPU processes RGB-D information from the depth camera to track user and calibrate FTVR system.

This work was supported by the Electrical and Computer Engineering department Human Computer Interaction lab at the University of British Columbia. The work presented in this paper was mainly contributed during the UBC course EECE570 Visual Computing instructed by Dr. Rafeef Garbi.
K. X. Author is a MASc student with the University of British Columbia, Vancouver, BC V6T 1Z4, Canada (email: zxia0101@student.ubc.ca).

recognition method by RGB-D information from a depth camera with major goals in fast detection speed, invariance to illumination, and detection in relatively far distance (beyond 2m). Standalone devices like FTVR create better user experience when the room is darker, so we aim to detect faces under variant level of illuminance. Robustness in relative longer distance is desirable due to the size scalability of FTVR and the potential for large number of users. The accumulative latency in hardware and software is one of the largest challenges in human computer interaction problems. Latency test results conducted by [5] have shown that even latencies of 50 and 100ms reduce the user experience in visually following objects and can be considered as a major fault for an interactive system. Due to the limitation to access lab during pandemic, this report focuses on software speed optimization and provides an alternative camera speed evaluation as reference.

Nowadays, face detection and recognition has become a rather mature technology that is applied to various applications like security camera, smart phone ID validation, and social media entertainment. It has attracted huge research attention from all over the world because face detection with high accuracy and robustness under critical conditions can solve tasks that may take a thousand or more times of effort for a human to solve. One recent extreme example by this technology came from China; Chinese authorities tracked down a fugitive among a 60,000 people crowd in a concert using face recognition [7]. Most standard face detection algorithms can detect faces with high accuracy within a short distance or with high quality pictures. But when occlusion and variation of faces occur, those methods create false positive or fail to detect any faces. With the primary objective of fast detection and general robustness in distance and illumination variation, we present a deep learning based framework taking RGB-D information as input to meet the calibration needs of our FTVR system.

II. Related Work

A. Traditional Face Detection and Recgonition

Face detection and recognition problem had been investigated back in early 1990s. The earliest method like EigenFaces recognition compares the eigen vector weights from known faces to the test faces after projecting the face images to a feature space [8]. It is easy to implement but ineffective for faces with variations and different lighting conditions. Viola Jones method is a major milestone in face detection. It combines the idea of Haar-like features, AdaBoost Algorithm and cascade classifier to achieve high detection accuracy and reasonable computational cost [9]. Haar-like features is a feature extraction method that matches a black and white box shaped in different arrangements to the target regions to generate matching scores for evaluation. Haar-like method has extremely high computational cost since each Haar box needs to slide through the whole image and computes at each individual pixel. AdaBoost is a machine learning based classifier that solved many difficult classification problems by maintaining distribution weight over the iterations of training samples [12]. AdaBoost uses extracted features to create weak and strong classifiers and forms a cascade classifier only by strong classifiers. The sequential cascade classifier rejects the low likelihood input regions at each stage to largely reduce the computation cost. But even with AdaBoost and cascaded structure, the computation power back at the time did not allow Viola and Jones to evaluate this method on the large and complex datasets. Despite the computation inefficiency, recent evaluation shows its accuracy performance is still comparable with Convolutional Neural Networks (CNNs) based techniques [13]. As the complex datasets like WIDER FACE become available, researchers realize that to account for the complex variations including pose, illumination, color, texture, and resolution, the existing detection algorithms take much longer computation time than expected. So, the new emerging challenge shifts to the trade-off between the computation efficiency of complex face variations and high accuracy [10]. The cascade structure used in Viola Jones inspired many other methods that focus on tackling the detection of face variations. The emergence of feature extraction methods like Scale Invariant Feature Transform (SIFT) and Histogram of Oriented Gradients (HoG) provide more accurate feature localization. SIFT applies a Laplacian filter at different scales to the target image to create a multi-scale extra pyramid. Subtraction between each level of pyramids generates Difference of Gaussian (DoG). By inspecting local neighborhoods in the DoG, the minima and maxima are detected as key points. After removing low contrast and poorly localized key points, histogram representations are computed based on the direction of the local gradient. By assigning descriptors to those local histogram representations, SIFT achieves high robustness in feature recognition at variant scale and affine transform level [11]. Both SIFT and Viola Jones methods have been adopted and further developed as the foundation of recent deep learning methods to overcome the challenges in detecting face variations and detection efficiency.

B. Face Detection and Recgontion by Deep Learning

More recently, face detection and recognition has been dominated by the deep learning methods due to its near perfect accuracy and the availability of annotated training data such as MegaFace and LFW [10]. The backbones of CNNs normally contains convolutional layer, activation function, pooling layer, fully connected layer, and logistic layer. By forming different combinations of those layers, and leveraging the output function, CNNs have been applied to tasks like classification, object detection, and object segmentation. The main reason of using deep learning on those tasks is that we can train deep CNNs with a large database so that the network learns to correctly output a category for the given input. Face detection is one the many object detection tasks that researchers have focused on since it creates significant applications. In general, the deep CNNs work by extracting images patches from the target image, passing the patches into the CNNs to predict if the patches include a face or not, and drawing bounding boxes on the patches that contain faces. The patch extraction process follows either the typical sliding window method or a more advanced approach called selective search presented in Region-based Convolutional Neural Network (R-CNN). Selective

search warps segmented regions, recursively combines similar regions by hierarchical grouping and produces the final region candidates to feed into CNNs [14]. But even with this selective approach, the network still takes more than 10s to process a single test image. Rather than generating candidate regions to feed into R-CNN, Fast R-CNN feeds the original image to R-CNN to generate a convolutional feature map. By using a region of interests (RoI) pooling layer and the same R-CNN classifier, Fast R-CNN achieves 10 times faster processing time than R-CNN [15].

Aside from high accuracy in face detection, various deep CNNs also have been developed to address issues on detection speed, face variations and detection distance. Research conducted in [16] trains the network with Long Distance heterogeneous Face Database (LDHF-DB), which contains both visible light and near infrared face images at distances of 60m, 100m, and 150m. SIFT is used as a basic feature extraction method and random discriminant subspaces are built to perform face recognition. This method demonstrates the importance of dataset selection for deep CNNs by showing long distance and night face detection results that outperforms the state-of-the-art commercial SDKs FaceVACS and PittPatt. Illumination change is tackled by a local binary pattern (LBP)-AdaBoost framework, which achieved accuracy of 97.27% on E-face dataset and 99.06% on XM2VTS dataset, respectively [17]. LBP generates binary representation of each pixel by comparing with its neighboring pixels. With LBP, the calculated image is resistant to fluctuations in gray scale values, thus invariant to illumination changes. However, Lee still uses the standard image pyramids to detect faces and eyes, which is the main reason that its detection speed only reaches 5.26 frames/sec. YOLO is the state-of-the-art accuracy deep CNN with fairly fast detection speed (45 frames per second) as well. Comparing with Fast R-CNN, which performs detections for many times with the given input regions, YOLO only passes the image through network once and output an (mxm) grid with bounding boxes and probabilities [18]. However, each grid cell only predicts two bounding boxes, which limits the prediction for objects that are spatially close. In YOLOv2, after finetuning with ImageNet, anchor boxes are added to predict the offsets of the bounding boxes rather than coordinates to increase the prediction feature refinement [19]. Even though the recall rate increases by 7% in the second version but the mean average precision drops slightly in the second version.

III. Methods

The pipeline in this work consists of two major components: face detection and face recognition. We started face detection by implementing Kanade-Lucas-Tomasi (KLT) and Multi-task Convolutional Neural Network (MTCNN). Some preliminary tests showed that MTCNN was not only faster but also more robust in longer distance detection than KLT. But even with MTCNN method, we were only able to achieve 315ms of delaying time and 80 lx minimum of mean illuminance value with the standard model. So, we started investigating the deep structure and parameters in MTCNN to further improve the detection performance in speed, illuminance and distance. The original MTCNN was trained on WIDER FACE and CelebA. We attempted fine tuning MTCNN with subsets of LDHF-DB to improve distance and illuminance performance, but due to VPN interruption, the training for more than 2 hours never succeeded. So, we used the weights trained on WIDER FACE and CelebA. Our face recognition system was built by transfer learning on VGG-16 with 3 live input videos.

A. MTCNN

Comparing with Viola Jones and AdaBoost method, MTCNN is more robust in real-world applications with large variations on human faces. Along with the cascaded structure, MTCNN takes advantage of the classifying task using convolutional neutral networks and regression task to achieve good performance and fast detection speed [6]. MTCNN is a deep neural network based face detection and facial landmark prediction method that can be used on images and videos. The general structure of MTCNN is formed by image pyramids and three cascading networks: Proposal Network (P-Net), Refinement Network (R-Net), and Output Network (O-Net). P-Net proposes all candidate windows by scanning at 12x12 resolution level and returns bounding boxes of the resulted faces to feed into R-Net. R-Net rejects a big portion of the false positive faces by increasing the receptive field of the input image. The returned bounding boxes from R-Net is passed to the O-Net to further refine the detected faces with predicted facial landmarks. Within each network, bounding box regression and non-maximum suppression (NMS) are also performed before the bounding boxes are outputted.

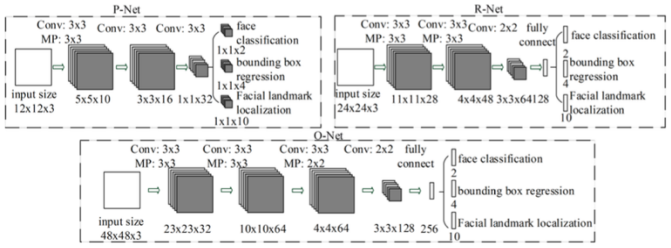

Fig. 2. MTCNN structure uses three CNNs: P-Net, R-Net, and O-Net. The number of layers in each network gradually deepens. The size of the input image increases from 12 to 24 to 48. The feature dimension of the output also increases from 32 to 128 to 256 [Zhang, 2016].

1) Face Classification

The face classification followed a typical two-class classification task format with cross-entropy loss:

$$L_i^{det} = -(y_i^{det} \log(p_i) + (1 - y_i^{det})(1 - \log(p_i))) \quad (1)$$

where p_i indicated the probability that the bounding box contained a face and y_i^{det} is the ground truth.

2) Bounding Box Regression

The bounding box calibration was achieved by transforming the output bounding boxes with the offset predicted by each network. This prediction model was trained by formulating a regression problem for regression target $y^o{}_i^{box}$ and the ground truth $y^{box}{}_i$ to minimize the Euclidean loss [6]:

$$L_i^{box} = \| y^o{}_i^{box} - y^{box}{}_i \|_2^2 \quad (2)$$

where $y^{box}{}_i$ is a 4-element vector that contains the box top left coordinates, height, and width. The facial landmark follows the

same loss function, but the facial landmark is a 10-element vector that contains coordinates for eyes, nose, and two mouth corners.

3) NMS

After bounding box regression, there are still multiple overlapping boxes. Similar to the approach used in [24], we first filtered out the boxes with confidence lower than 0.2. Then the nearby boxes were grouped into clusters to compare the confidence score. The window with the maximum confidence score was used to filter out the other windows with less than 80% of the confidence score. The remaining windows were then averaged in terms of location to get the final output window at each cluster position.

4) Training Data Annotation

The training data was categorized into four kinds based on the Intersection-over-Union ratio:

a) Negatives: image area with ground-truth faces IOU < 0.3, label = 0. b) Positives: image area with ground-truth faces IOU >0.65, label = 1. c) Part faces: image area with groudn-truth faces 0.4<IOU<0.65, label = -1. d)Landmark faces: images with 5 key points, label = -2.

B. Face Detection by Depth Information

1) Noise Filtering

Comparing with RGB images, the depth information provides an accurate description of object locations which can be used to support face detection with variations in illuminance and distance. The depth information is presented as 16bits 2D depth map. Each cell value represents its relative distance based on the focal length of the depth camera. However, the quality of depth map is degraded by mixed pixels, lost pixels, and noisy pixels due to the non-linear behavior of IR sensors and the reflectivity pattern on different object surface [21]. To reduce the impact of noise, we apply a simple and effective 3x3 median filter to the whole image.

2) Human Extraction

Under the FTVR setting, we have access to a static background information before the user starts to interact with the system. This background information provides great advantage to segment the user from all pre-existing objects. We averaged the first 30 frames of the live video input to account for any subtle change in the background and create the "Smoothed Environment" frame. From each input frame, we simply subtracted smoothed environment frame to generate the human location in the scene.

3) Face Detection

One of the biggest advantages by using depth information for face detection is that the size of the faces in the depth map can be estimated so that the typical image pyramids are not necessary to find the faces of different sizes [22]. We used a square rather than a rectangle to capture faces for simplicity and coherence with later detection method. The face candidate window size S can be calculated by the distance D_p, the actual face size R (200mm), and the focal length of the camera f according to equation (3):

$$s = \frac{f \cdot R}{D_p} \tag{3}$$

Depth map analysis from Figure showed that for a canonical view face, the nose position corresponded with a local minimum. Depending on the shape of different individuals, the shape of the local minimum could differ but could be characterized either by pre-calibration. Based on this idea, we measured the flatness by calculating the relative peak value and the standard deviation of all cells at each candidate window. The relative peak value and the standard deviation for a user was manually calculated and used as the template to assign a probability score p to each candidate window. Those candidate windows were first filtered by a threshold value against the probability score and then fed into the MTCNN network for further processing.

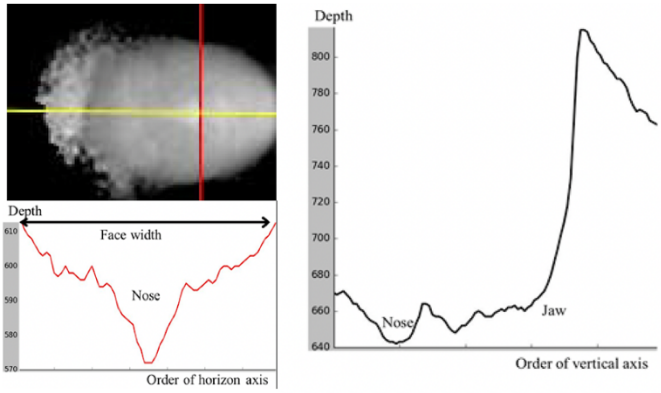

Fig. 3. (top left) Sampled face depth map, (bottom left) Depth across the horizontal axis marked by red line, (Depth across the vertical axis marked by yellow line) [25]

C. Proposed Face Detection Structure

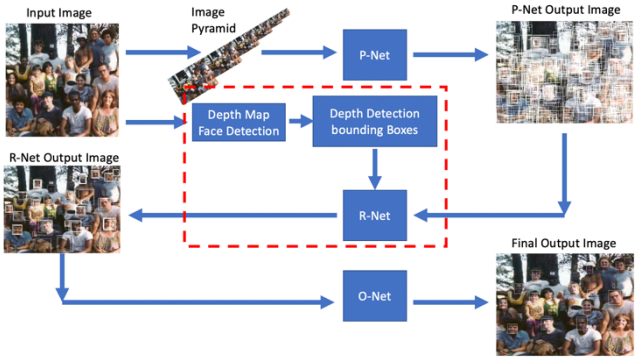

Fig. 4. MTCNN + depth map face detection pipeline. R-Net takes additional bounding box input from the depth map prediction (included in the dotted red box) comparing to the original MTCNN structure.

Combining the MTCNN structure and the depth detection, we propose the face detection structure as shown in Figure 4. The depth camera provides supplementary bounding boxes for the conditions where MTCNN is not robust enough for long distance and low illumination face detection. The bounding box regression in R-net also helps to reduce the false negative predicted by the depth map method.

D. Face Recognition by Fine Tuning VGG-16

1) VGG-16

VGG-16 won the ImageNet competition 2014 and is known for its great performance in object classification. VGG-16 only uses convolutional layers of 3x3 with stride of 1 and padding and maxpool layer of 2x2 with stride of 2. The same structure is repeated throughout the whole architecture. There are 16 layers in the network that contains weight, which is why it is called VGG-16. It takes a fixed input image size of 244x244x3 and uses softmax to output which class the image belongs to.

2) Transfer Learning

Fine tuning is a special type of transfer learning that only trains certain part of the deep CNNs to perform another task. For our face recognition task, we used VGG16 as the backbone and performed fine tuning with 3 user input live video to mimic the situation of using our FTVR: user needs to stand in front of the camera for 10 seconds and wait for the projector calibration and face recognition training.

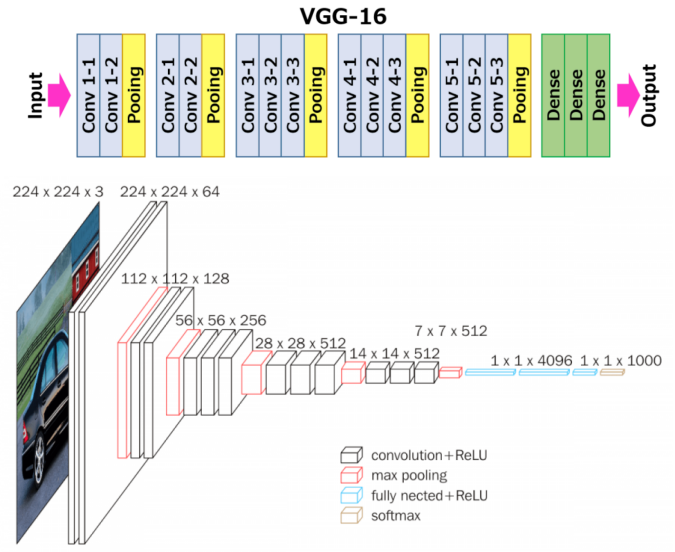

Fig. 5. VGG-16 Framework

IV. Project Summary and Contributions

We implemented a face detection and face recognition system and performed test on detection speed, illuminance variation, and distance variation for our system. The face detection system was built on top of the (MTCNN). With Github resources, the MTCNN was thoroughly studies and the main structure (image pyramid, and P, R, O-Net) was reimplemented and integrated with depth information input. The depth map face detection structure was inspired by the ideas mentioned in the methods section, but the implementation was developed from scratch. The face recognition transfer learning was guided by online resources, but the learning parameter and implementation was tuned and modified for our own use. Our major contribution lies in implementing a robust face detection system and recognition system by leveraging existing networks (MTCNN and VGG-16) and our own network (Depth map face detection).

V. Materials and Implementation Details

A. Hardware

UBC HCI lab had purchased an Intel RealSense D455 for the research use. But due to the pandemic impact and limitation to the lab, we purchased another camera with similar performance as the D455 for the data acquisition in this experiment: Astro Pro Kinect LeTMC-520 (Figure 6). Its specifications are listed in Table 1.

RGB Maximum Frame Rate	30FPS
Depth Maximum Frame Rate	30FPS
Image Sensor and Depth Sensor	MT9M001 + AR0330
Depth FOV (H x V)	60º x 46.7º
Maximum Depth Resolution	640 x 480 (pixel x pixel)
Maximum RGB Resolution	1280 x 720 (pixel x pixel)

Table 1: Astro Pro Kinect LeTMC-520 Specifications

Figure 6: (Left) Intel RealSense D455 camera located in UBC HCI lab. (Right) Astro Pro Kinect LeTMC-520 camera, used for this experiment

B. Software

The main processing software used in the experiment for face detection and face recognition was Google Colab under the python programming environment with the support of computer vision libraries: cv2, matplotlib, scipy, pytorch, panda, IPython etc. Some test results and plots were generated by Matlab.

The main software used for interfacing with the depth camera was OpenNI 2 and Visual Studio under the C++ programming environment.

C. MTCNN

The implementation of MTCNN was guided by multiple Github resources and the original paper [6]. The mainly used ones are mtcnn-pytorch:https://github.com/TropComplique/mtcnn-pytorch/blob/master/src/box_utils.py and facenet-pytorch: https://github.com/timesler/facenet-pytorch.git. We reintegrated the three stages of the network for the easy use of our depth map bounding box input. But we used the bounding box utilities, network weights, and some detail level implementation. More details can be found in code file.

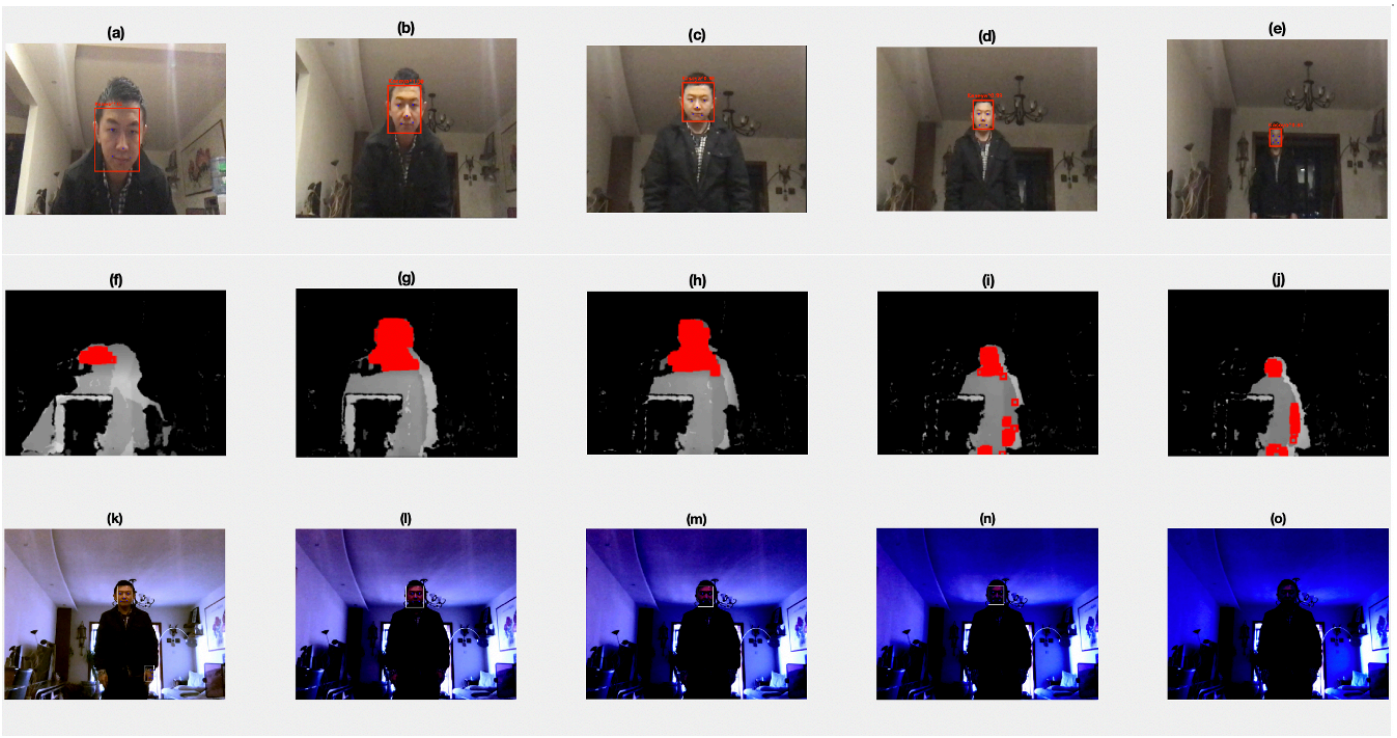

Fig. 7. (a)-(e) The first row shows the distance test at 0.5m, 1m, 2m, 3m, and 4m. (f)-(j) The second row shows the depth map distance test at 0.5m, 1m, 2m, 3m, and 4m. (k)-(o) The third row shows the illuminance test with the image mean illuminance value at 109, 69, 51, 35 and 25.

D. Transfer Learning by VGG-16

Transfer learning was guided by online resources but the parameters were modified for our use and the data live video input data directory was built from scratch: https://towardsdatascience.com/finetune-a-facial-recognition-classifier-to-recognize-your-face-using-pytorch-d00a639d9a79 and https://medium.com/@chiraggoelit/face-recognition-using-transfer-learning-9986728c443d.

The 3 user input videos were captured into frames and cropped to VGG-16 input size, which was 224x224. MTCNN was used to detect faces and align the faces to the center of the cropped image. 100 samples were randomly picked from each user image pool to diversify the face pose. The data were then split into training and validation folders with 8:2 ratio; the separated data directory is shown in Figure 7. The training data were then augmented by 45 degrees rotation, 0.3-pixel linear shift and horizontal flipping. The training model hyperparameters were learning rate of 0.001, epochs of 5, mini-batch size of 16.

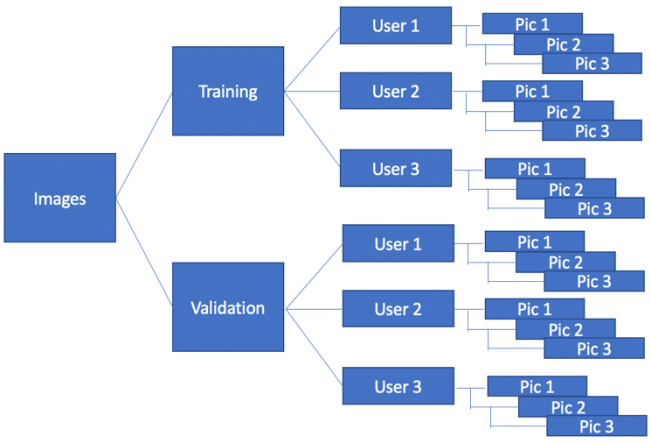

Figure 7: Training data directory

E. Depth Data Acquisition

The depth camera was connected to a PC via USB 3.0 cable connection and driven by Astro Pro driver. We developed code to extract RGB and depth videos from frame buffer based on the template code provided by Astra Orbbec SDK, which can be found here: https://orbbec3d.com/develop/. The exported video from the frame buffer was in Oni. format, an example frame for the video was shown in Figure 8. The video was uploaded to Google Colab and extracted frames by

pyOniExtractor, which was a GitHub code based on OpenNI 2. We modified the code so that the output frames were in the numbered order for the convenience of our preprocessing.

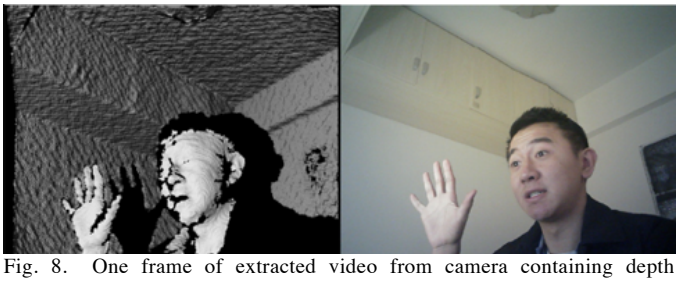

Fig. 8. One frame of extracted video from camera containing depth information on the left and RGB on the right

After applying 3x3 median filter and extracting smoothed environment frame, the input image was fed into face detection algorithm.

VI. Results

A. Distance Test

We performed distance test with RGB and depth input at 0.5m, 1m, 2m, 3m, and 4m. The depth map test failed to detect the proper size of bounding boxes. So, feeding those tiny bounding boxes into R-Net resulted in 0 bounding box returns after regression. Thus, we had to present the RGB detection and depth detection results separately. (a)-(e) showed the RGB test results, which demonstrated robustness up to 4m. (f)-(j) showed the depth map detection results. For distance within 2m, the bounding boxes were able to localize the head area, but not with proper size. The focal length parameter, threshold value, the template peak value and template standard deviation value were adjusted but no major improvement was observed.

B. Illuminance Test

(k)-(o) showed the illuminance test results with mean illuminance value of 109, 69, 51, 35 and 25, respectively for the input image. The model was able to detect faces in images with a minimum mean illuminance value of 35.

C. Speed Test

Task	Camera Data Acquisition	MTCNN Face Detection	Depth Map Face Detection	Face Recognition
Speed (ms)	350	134	170	250

D. Transfer Learning Accuracy

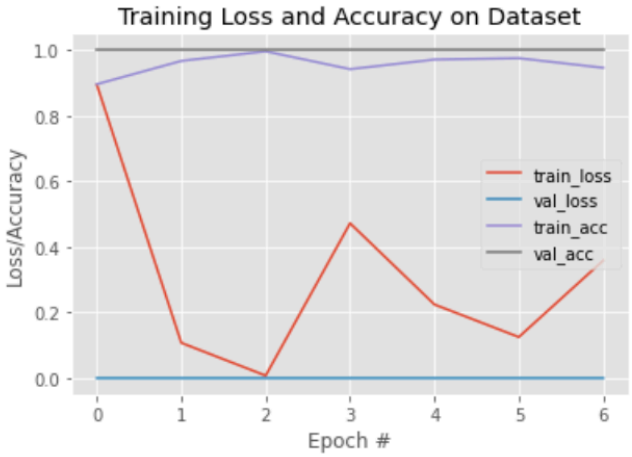

Fig. 9 Transfer learning results on VGG-16 with 100 random samples from 3 user live video input

Transfer learning on VGG-16 with 3 classes of users and each class containing 100 random samples from a live video results were shown in Figure 9. Both validation accuracy and training accuracy reached a value that was very close to 1. Both training loss and validation loss reduced to below 0.4. The training of 6 epochs only took around 24s, which made it feasible to train the model on site with a new user for FTVR. Further testing on live feeding videos found that a training epoch of more than 10 caused the model to overfit so that it heavily recognized the first user class rather than the other two. We tested the scenery where one trained user was present in the view, the model was able to track and recognize the user for 80% of the time. We also tested the condition that when an unknown user walked into the view where a known user was present. Due to the lack of the user live video annotation, this test was conducted by exporting all the predicted frames and combining into video to visually inspect the detection label. The algorithm was not robust enough to separate the unknown user and the known user and generated false positive rate to as high as 40%. Figure 10 shows some frame success and failure examples in user recognition.

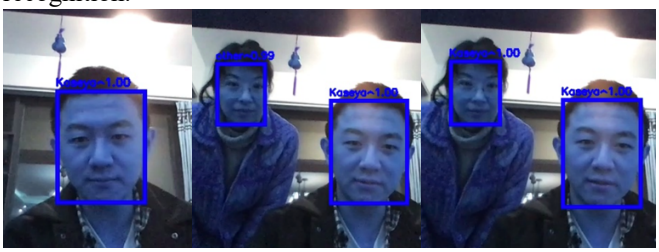

Fig. 10. Face recognition test frame examples for trained user "Kaseya". (Left) Success example for known user recognition. (Mid) Failure example for one known user and one unknown user recognition. (Right) Success example for one known user and one unknown user.

VII. Conclusions

In this paper, we presented a face detection and face recognition system for FTVR and 3D teleconferencing application. Combining existing deep networks and our proposed depth map detection method, we implemented our system with the goal in

fast detection speed, invariance to illumination and detection distance. Even with the failure in depth map method, the MTCNN network was still able to achieve invariance to illumination and distance. The detection speed was not as fast as expected but as the test showed, the system latency came in from different parts. To optimize the detection speed, resolution, detection accuracy and other metrics needed to be sacrificed. Frame buffer management was the part we did not have time to investigate to increase the overall system speed. The biggest regret from this work was that we put a lot of time into understanding deep learning but due to the VPN limitation, we were not able to train our network with large datasets.

Future work includes: 1) rework the depth map to generate proper size bounding boxes and more relevant confidence score. 2) Optimize MTCNN detection speed by fine tuning the model parameters and test with large datasets. 3) Train and test our system on large datasets to provide more comparable results with literature methods.

REFERENCES

[1] Fafard, Dylan & Stavness, Ian & Dechant, Martin & Mandryk, Regan & Zhou, Qian & Fels, Sidney, 2019. FTVR in VR: Evaluation of 3D Perception With a Simulated Volumetric Fish-Tank Virtual Reality Display. CHI '19: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. 1-12. 10.1145/3290605.3300763.

[2] Q. Zhou, G. Miller, K. Wu, D. Correa and S. Fels, "Automatic Calibration of a Multiple-Projector Spherical Fish Tank VR Display," 2017 IEEE Winter Conference on Applications of Computer Vision (WACV), Santa Rosa, CA, 2017, pp. 1072-1081, doi: 10.1109/WACV.2017.124.

[3] Kuster, Claudia & Ranieri, Nicola & Agustina, Agustina & Zimmer, H. & Bazin, J.C. & Sun, C. & Popa, T. & Gross, M, 2012. Towards next generation 3D teleconferencing systems. 3DTV-Conference. 1-4. 10.1109/3DTV.2012.6365454.

[4] Cipresso, P., Giglioli, I., Raya, M. A., & Riva, G. (2018). The Past, Present, and Future of Virtual and Augmented Reality Research: A Network and Cluster Analysis of the Literature. *Frontiers in psychology*, *9*, 2086. Doi: 10.3389/fpsyg.2018.02086

[5] Jota, Ricardo & Ng, Albert & Dietz, Paul & Wigdor, Daniel. (2013). How fast is fast enough? A study of the effects of latency in direct-touch pointing tasks. Conference on Human Factors in Computing Systems - Proceedings. 2291-2300. 10.1145/2470654.2481317.

[6] K. Zhang, Z. Zhang, Z. Li and Y. Qiao, "Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks," in *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499-1503, Oct. 2016, doi: 10.1109/LSP.2016.2603342.

[7] N. Louise, "Chinese man caught by facial recognition at pop concert", *BBC News*, 2020.

[8] M. Turk & A. Pentland"Eigenfaces for recognition", Journal of Cognitive Neuroscience, 2020.

[9] Viola, Paul & Jones, Michael. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE Conf Comput Vis Pattern Recognit. 1. I-511. 10.1109/CVPR.2001.990517.

[10] Shepley, Andrew. (2019). Deep Learning For Face Recognition: A Critical Analysis.

[11] Lowe, David. (2004). Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision. 60. 91-. 10.1023/B:VISI.0000029664.99615.94.

[12] Y. Freund & R. Schapire, "A brief introduction to boosting | Proceedings of the 16th international joint conference on Artificial intelligence - Volume 2", 2020.

[13] Li, Haoxiang & Lin, Zhe & Shen, Xiaohui & Brandt, Jonathan. (2015). A convolutional neural network cascade for face detection. 5325-5334. 10.1109/CVPR.2015.7299170.

[14] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.

[15] R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.

[16] Maeng, Hyunju & Liao, Shengcai & Kang, Dongoh & Lee, Seong-Whan & Jain, Anil. (2012). Nighttime Face Recognition at Long Distance: Cross-Distance and Cross-Spectral Matching. 708-721. 10.1007/978-3-642-37444-9_55.

[17] Lee, Hansung. (2020). Face Recognition at a Distance for a Stand-Alone Access Control System. Sensors. 20.

[18] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[19] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6517-6525, doi: 10.1109/CVPR.2017.690.

[20] S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), Kuala Lumpur, 2015, pp. 730-734, doi: 10.1109/ACPR.2015.7486599.

[21] S. Kim, M. Kim and Y. Ho, "Depth image filter for mixed and noisy pixel removal in RGB-D camera systems," in IEEE Transactions on Consumer Electronics, vol. 59, no. 3, pp. 681-689, August 2013, doi: 10.1109/TCE.2013.6626256.

[22] Ballotta, Diego & Borghi, Guido & Vezzani, Roberto & Cucchiara, Rita. (2018). Head Detection with Depth Images in the Wild. 56-63. 10.5220/0006541000560063.

[23] Peng, Xiaoming & Bennamoun, Mohammed & Mian, Ajmal. (2011). A training-free nose tip detection method from face range images. Pattern Recognition. 44. 544-558. 10.1016/j.patcog.2010.09.015.

[24] Farfade, Sachin & Saberian, Mohammad & Li, Li-Jia. (2015). Multi-view Face Detection Using Deep Convolutional Neural Networks. 10.1145/2671188.2749408.

[25] Kim Heung-jun, Lee Dong-seok, Kwon Soon-kak. Implementation of Nose and Face Detections in Depth Image. J Multimed Inf Syst 2017;4(1):43-50.
https://doi.org/10.9717/JMIS.2017.4.1.43