Face Depth Estimation and 3-D reconstruction

Alba Terese Baby, Aleesha Andrews, Amal Joseph, Amal Dinesh
and V. K. Anjusree

April 20, 2020

Face Depth Estimation and 3-D reconstruction

Alba Terese Baby[1] , **Aleesha Andrews**[2] , **Amal Joseph**[3] , **Amal Dinesh**[4] , **Anjusree V.K**[5]
[1,2,3,4,5] Department of Computer Science and Technology,
Kerala Technological University,
Rajagiri School of Engineering and Technology , Rajagiri Valley,
Kakkanad.
[1,2,3,4] U.G students,
[1]albaterese30@gmail.com, [2]aleeshandrews57@gmail.com, [3]amaljoseph@gmail.com,
[4]amalkdinesh@gmail.com
[5] Assistant Professor,
anjusreevk@rajagiritech.edu.in

Abstract

In the world of fast growing technology people look for more realistic representation and hence 3D representation of 2D images acquire great importance. 3D models are used in various fields like face recognition and animation games. They are widely used in medical industry to create interactive representations of anatomy. However, constructing 3D models or reconstructing from 2D images have been a major challenge for the researchers. Many approaches have been proposed and developed for generating 3D representation. In this work, we developed a Generator Adversarial Network(GAN) based method for depth map estimation from any given single face image.Here we have used pix2pix which is a variant of the conditional GAN.It is capable of performing image-to-image translation using the unsupervised method of machine learning.We found that it is the most robust method.

I. Introduction

Face depth estimation and 3-D reconstruction has been a challenge for many researchers.But with the advent of Generative Adversarial Network(GAN) based method all the barriers faced earlier were resolved.GAN based methods are an unsupervised learning method.Manually acquiring labeled data requires a lot of time but GAN doesn't require labeled data.They train unlabeled data as they learn the internal representation of data.Another major advantage of GAN based model is that they generate output similar to the real data. Because of this, they have many different uses in the real world. They can generate images, text, audio, and video that is indistinguishable from real data.

Our aim is to implement a GAN based method for depth map estimation from any given single face image.Depth estimation is a computer vision task designed to estimate depth from a 2D image.3D reconstruction helps in overcoming issues related with 2D images and helps to improve accuracy in various tasks.Many works have been proposed over few decades and finally we conclude that a conditional GAN based solution is the most robust approach.

One variant of GANs is named as Conditional Generative Adversarial Networks (cGAN)[5].Conditional GANs (cGAN) incorporate input data as a conditioning variable.In cGAN both the generator and discriminator are conditioned on some extra information.This could be any kind of auxiliary information, such as class labels or data from other modalities.Motivation behind cGANs is to tackle the image-to-image transformation problems. The difference of a cGAN from an unconditional network is that input images are fed to both discriminator and generator networks. Some of the application areas of cGANs have been background masking, segmentation, and interesting implementations such as edges-to-objects.

Our primary objective is to train the GAN model to generate 3D depth map of a given 2D image. Pix2pix GAN, a variant of cGAN is choosen as the suitable network architecture in this work.As any GAN based model our model has also got a generator and a discriminator.Generator generates image from the input and discriminator computes how close that generated image is to the reference image.Discriminator does this using L1 loss function[13].It is the sum of all absolute differences between the true value and the predicted value.

II. Related Works

A. Shape from shading

Shape from shading (SFS)[1][2] uses the pattern of shading in a single image to acquire the shape of the surface in view. A typical example of shape from shading is astronomy, where the technique is used to reconstruct the surface of a planet from photographs acquired by a spacecraft. The reason that shape can be generated from shading is the link between image intensity and surface slope. The radiance at an image point can be calculated with the surface normal, the direction of the illumination (pointing towards the light source) and the albedo of the surface, which is typical of the surface's material. After calculating the radiance for each point reflectance map of the image is obtained. The parameter of the reflectance map might be unknown. In this case the albedo and illuminant direction is to be obtained. Albedo and illuminant can be computed, by looking at a lambertian surface, with help of the averages of the image brightness and its derivatives. From the reflection map and by assuming local surface smoothness, can estimate local surface normal, which can be integrated to give local surface shape.

Smooth variations in the brightness or shading of objects in an image are often an important cue in human vision for estimating the shape of depicted objects. This project is concerned with modelling this process and setting up prototype systems for automatically recovering surface shape from image shading.The results can potentially be put-in to the computer interpretation of satellite, medical and Synthetic-aperture radar images, as well as automated visual inspection of industrial parts.

B. Auto Encoders

The basic idea is to learn the subspaces of both 2D sample images and 3D sample faces, model the mapping function from the 2D subspace to the 3D subspace based on the low-dimensional features, then link the 2D subspace learner, the mapping function and the 3D subspace learner together to form a deep framework which takes 2D images as input and 3D models as output. To this end, the subspace learner should be invertible which means it can recover original data from the low-dimensional features exactly.

A nonlinear Stacked Contractive Autoencoders (SCAE)[9] as subspace learner to take-out low-dimensional features more robust to trivial variations of training data, and exploit a one-layer fully connected neural network to build the mapping function. Given the pre-trained parameters of the SCAEs, it is easy to initialize a deep feedforward neural network to model the reconstruction process from 2D image to 3D face. Owing to the good initialization, the training of the network should be fast and the reconstruction should be accurate.

III. Network Structure

The Generative Adversarial Network[4] used a noise variable as input, where as conditional GANs (CGAN)[5] incorporate input data as a conditioning variable. This conditioning has been applied in many applications, such as labels, text, images,videos and also in general non application-specific structure.The Conditional GAN structure used for the depth estimation goal can be defined as follows: Let G and D represent two networks, generator and discriminator, respectively. G maps a random Gaussian noise z under the condition of observed image x to depth map d:

$$G:\{x,z\} \rightarrow d$$

In training the generator network, our aim is to maximize the objective function

$$L_G(G;D)= \sum \log D(x;G(x;\ z))\ (1)$$

where G tries to force D to accept generated depth maps as true outputs. At the same time D is trained to discriminate fake maps from real ones, maximizing the objective function:

$$L_D(G;D)= \sum \log D(x;\ d) + \log(1-D(x;G(x;\ z)))\ (2)$$

The first part of the last equation represent the training with real images to real depth maps, while second part covers the output maps of the generator network, labeled as fake. An additional distance loss term can be added in Equation 1 to prevent the generator from moving too far away from the ground-truth data during the training process. This term can be an L2 distance loss[13](Sum of the squared differences between the true value and the predicted value), or an L1 distance loss[13] (Sum of the absolute differences between the true value and the predicted value).

The final objective function for the generator can be written as:

$$G=\arg\min\max L_G(G,D) + \lambda d_{L1|L2}(G)\ (3)$$

where L_G is the loss function given in Equation 1. The last term is a L1- or a L2-norm distance function.

Training GANs have been reported to be problematic for many reasons, including non-convergence where the model oscillates or never converges, vanishing or exploding gradients where the discriminator network overwhelms over the generator in the zero-sum game, mode collapses where the generator network does not learn and generates small number of outputs, and overfitting issues.

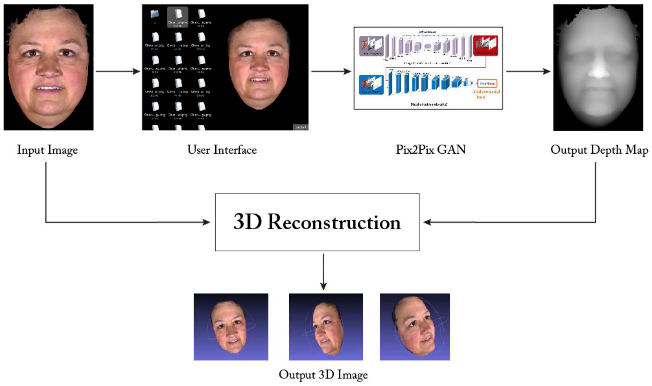


Fig. 1: General structure of a conditional generative adversarial network with inputs of 2D images and depth maps.

Pix2pix[6] is a GAN based method for image-to-image translation using conditional adversarial networks. Given a training set which contains pairs of related images (A and B) a pix2pix model learns how to convert an image A to an image B, or vice-versa.Main advantage of pix2pix is that it is generic.It does not require pre-defining the relationship between two types of images.Pix2pix is more flexible and adaptable than any other GAN based method. Here in our project we trained our pix2pix network to generate depth map of the given 2-D image.Once the network is trained it is capable of generating depth map from any given 2-D image as shown in Figure 1.

IV. Proposed Method

A. Network Architecture

Conditional GANs (CGAN) incorporate input data as a conditioning variable.Here we have used a pix2pix GAN which is a conditional GAN.As any GAN it has also a generator and discriminator.Generator generates an image and discriminator discriminates the generated image and the reference image.Here we have used U-net as generator and patch-GAN as discriminator.

a. Generator

Our model uses a U-Net[7] architecture for generator as shown in Figure(2)(3).The U-Net architecture is built upon the Fully Convolutional Network.The main difference between autoencoders and U-net is that U-net provides skip connections between the downsampling path and the upsampling path.This connections applies a concatenation operation instead of sum.These skip connections intend to provide local information to the global information while upsampling. The generator has a :-

- Encoder:- Encoder is the downsampling path.Here the size and number of layers of the inputed image is reduced.It consist of convolution layer,batch normalization and ReLU.
    1) Convolution:- Convolution is one of the main building blocks of a CNN. The convolution refers to the mathematical combination of two functions to produce a third function. It merges two sets of information.The convolution is performed on the input data with the use of a filter to produce a feature map.Feature map is obtained by sliding the filter all over the input.At every location,a matrix multiplication is performed and sum of the result onto the feature map.
    2) Batch Normalization:- Batch normalization is done to increase the efficiency of the network.It increases the speed of the network.Wide ranges of values are normalized to similar values so that the network speed increases.
    3) ReLu Function:- ReLU is an activation function. In ReLU all negative values from feature map is mapped to 0 and for all positive values input is taken as output.
- Decoder:- Decoder is the upsampling path.Here the downsampled image will regain its original size.It consists of transposed convolution,batch normalization,dropout.
    1) Transposed Convolution:- The need for transposed convolutions generally arises from the desire to use a transformation going in the opposite direction of a normal convolution, i.e., from something that has the shape of the output of some convolution to something that has the shape of its input.
    2) Dropout:- Dropout is a technique used to prevent a model from overfitting.Overfitting happens when a model learns the details in the training data and negatively impacts the performance of the model on new data.

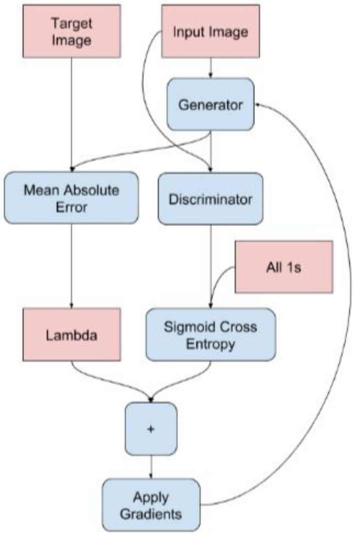


Fig. 2: Training procedure of Generator

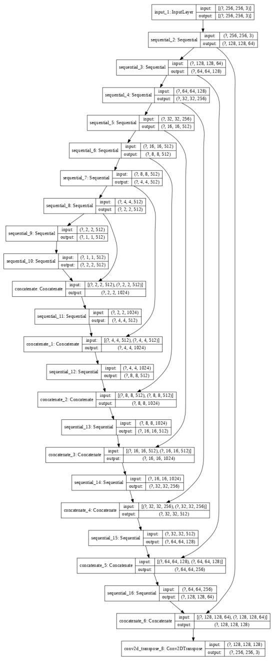

input_1: InputLayer | input: [(?, 256, 256, 3)] | output: [(?, 256, 256, 3)]
sequential_2: Sequential | input: (?, 256, 256, 3) | output: (?, 128, 128, 64)
sequential_3: Sequential | input: (?, 128, 128, 64) | output: (?, 64, 64, 128)
sequential_4: Sequential | input: (?, 64, 64, 128) | output: (?, 32, 32, 256)
sequential_5: Sequential | input: (?, 32, 32, 256) | output: (?, 16, 16, 512)
sequential_6: Sequential | input: (?, 16, 16, 512) | output: (?, 8, 8, 512)
sequential_7: Sequential | input: (?, 8, 8, 512) | output: (?, 4, 4, 512)
sequential_8: Sequential | input: (?, 4, 4, 512) | output: (?, 2, 2, 512)
sequential_9: Sequential | input: (?, 2, 2, 512) | output: (?, 1, 1, 512)
sequential_10: Sequential | input: (?, 1, 1, 512) | output: (?, 2, 2, 512)
concatenate: Concatenate | input: [(?, 2, 2, 512), (?, 2, 2, 512)] | output: (?, 2, 2, 1024)
sequential_11: Sequential | input: (?, 2, 2, 1024) | output: (?, 4, 4, 512)
concatenate_1: Concatenate | input: [(?, 4, 4, 512), (?, 4, 4, 512)] | output: (?, 4, 4, 1024)
sequential_12: Sequential | input: (?, 4, 4, 1024) | output: (?, 8, 8, 512)
concatenate_2: Concatenate | input: [(?, 8, 8, 512), (?, 8, 8, 512)] | output: (?, 8, 8, 1024)
sequential_13: Sequential | input: (?, 8, 8, 1024) | output: (?, 16, 16, 512)
concatenate_3: Concatenate | input: [(?, 16, 16, 512), (?, 16, 16, 512)] | output: (?, 16, 16, 1024)
sequential_14: Sequential | input: (?, 16, 16, 1024) | output: (?, 32, 32, 256)
concatenate_4: Concatenate | input: [(?, 32, 32, 256), (?, 32, 32, 256)] | output: (?, 32, 32, 512)
sequential_15: Sequential | input: (?, 32, 32, 512) | output: (?, 64, 64, 128)
concatenate_5: Concatenate | input: [(?, 64, 64, 128), (?, 64, 64, 128)] | output: (?, 64, 64, 256)
sequential_16: Sequential | input: (?, 64, 64, 256) | output: (?, 128, 128, 64)
concatenate_6: Concatenate | input: [(?, 128, 128, 64), (?, 128, 128, 64)] | output: (?, 128, 128, 128)
conv2d_transpose_8: Conv2DTranspose | input: (?, 128, 128, 128) | output: (?, 256, 256, 3)

Fig. 3: Architecture of U-Net.

b. Discriminator

The Discriminator used in our model is patchGAN as shown in Figure (4)(5).PatchGAN maps from a 256x256 image to an NxN array of outputs,where each output signifies whether the corresponding patch in the image is real or fake. We are using 70x70 patchGAN as discriminator network in our work.The model takes two images,specifically an input image and generated image,which it should classify as real.These images are concatenated together at the channel level.The shape of the output after the last layer is $(batch_{s}ize, 30, 30, 1)$ i.e each 30x30 patch of the output classifies a 70x70 portion of the input image. Each block in discriminator consist of convolution,batch normalization and Leaky Relu.

1) Batch Normalization:- Batch normalization helps in increasing the speed at which networks train.
2) Leaky ReLu Function:- Leaky ReLU fixes the dying problem as it doesn't have zero slope parts.The slope of the Leaky ReLU is set to 0.2.

The kernel size is fixed at 4x4 and a stride of 2x2 is applied.The number of filters will increase from 64 to 128, 256 and 512 as it passes through each Convolution layers.As the number of filters increases,the image size get reduced.The input size of 256x256 finally reduces to 30x30.

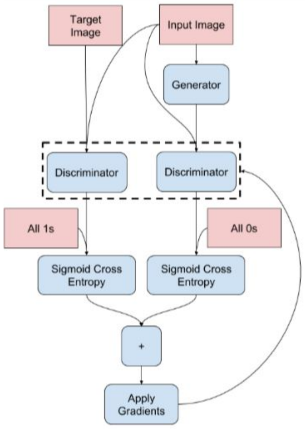


Fig. 4: Training procedure of Discriminator

B. Loss Function

Machines learn by means of a loss function. It is a method of evaluating how well specific algorithm models the given data. If predictions deviates too much from actual results, loss function would be a very large number.With the use of some optimization function, loss function learns to reduce the error in prediction.In our model we have used Mean Absolute Error(MAE) loss functions[13].

a. Generator Loss

The generator loss is the sigmoid cross entropy loss[14] of the generated images and array of ones.We have also used L1 loss function[13] which is MAE which is the sum of absolute difference between target image and generated image.This predicts how close the generated image is to the target image.

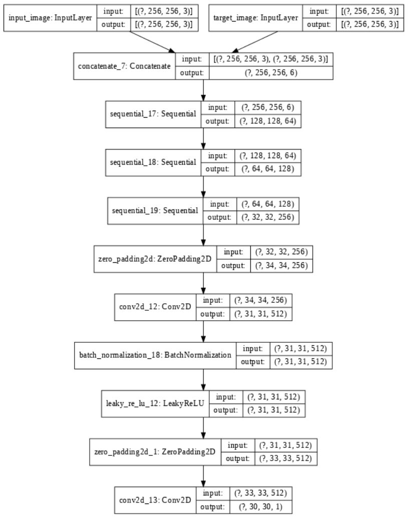


Fig. 5: Architecture of Patch GAN.

The formula to calculate the total generator loss is:

$$\mathrm{L}_{Gen}= Gan\_loss + LAMBDA*L1\_loss \ (4)$$

Here LAMBDA is set as 100.

$$\mathrm{MAE}=(\frac{1}{n})\sum_{i=1}^{n}|y_i - x_i| \ (5)$$

where y_i is the prediction and x_i is the true value.

b. Discriminator Loss

The discriminator has two inputs real images and generated images.Real_loss is a sigmoid cross entropy loss of the real images and an array of ones.Then the Total_loss is the sum of Real_loss and the Generated_loss.

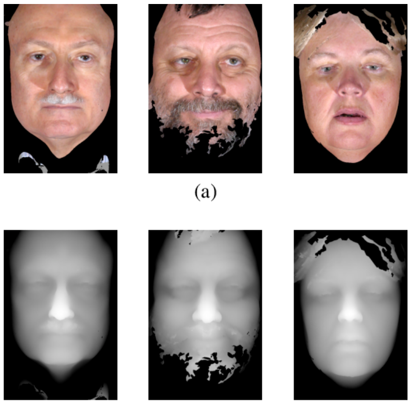
$$\text{Total \_ loss=Real \_ loss+Generated \_ loss (6)}$$

C. Experiments

In this section we focus on our experimental setup for training and testing,as well as the datasets used in our work.We have implemented a GAN based solution for depth map estimation and from that depth map we reconstructed the corresponding 3D image.Depth map is in gray scale format that contains information about the distance between the surface of objects from a given viewpoint. A small set of images are selected from the database after training of network for testing purpose.

D. Databases

The 3D face database that have been used in our work is Texas 3D Face Recognition Database[8] as shown in Figure 6.It consists of 116 individuals with varying poses and emotional expressions.In order to increase variance of the data set and improve robustness of the trained networks, augmentation can be put-on to the training data.Ground truth depth maps are provided along with the database.All input images and depth maps were resized to 256x256 resolution



(a)

(b)

Fig. 6: Texas 3-D Face Recognition Database.

since there are large background regions in the original images. 10 individuals from the database are randomly selected for testing purposes, and training data set is taken from the remaining image-depth pairs. The number of images used for training is 919 and for testing is 230.

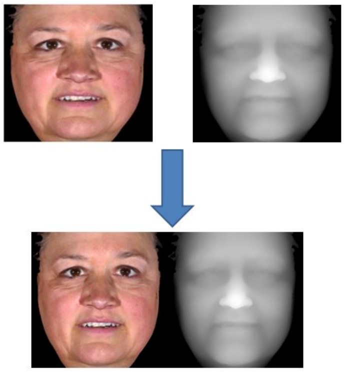


Fig. 7: Pre-processed stage

E. Training Details

All our training is done on colab.The machine learning framework we have used is Tensorflow[10].We had run the model for 90 epochs.Open3D which is a python module was used for 3-D reconstruction.Implementation of userinterface was done using a python module called kivy.

F. Setup

In this experiment we have used a pix2pix GAN as mentioned in the network architecture above.First, we apply random jittering and mirroring to the training dataset.In random jittering,the image is resized to 286*286 and then arbitrarily cropped to 256*256 and in random mirroring, the image is randomly flipped horizontally i.e left to right. Resized image and the range image are combined to a single image and given to the network for training as shown in Figure 7.

Preprocessed stage is followed by training and testing of the model.Our database are split for both the training and testing purposes.The generator generate the depth map of the input image. The discriminator receives the input image and the generated image as the first input. The second input is the input image and the target image.We calculate the generator and the discriminator loss.Then, we calculate the gradients of loss with respect to both the generator and the discriminator inputs and apply those to the optimizer.

Next step is to generate 3-D representation from depth or range image.For that purpose point cloud representation is created using the depth map generated from the model and the original image.A wireframe model is created from point cloud representation.Finally, the wireframe model is converted into proper 3D representation.A method called Poisson surface reconstruction[15] is used to reconstruct the face surface.It generates very smooth surfaces that robustly approximate noisy data.

We have implemented an user interface(Figure 8) for our model which is implemented using a module in python called kivy.Interface has single window for selecting the file.

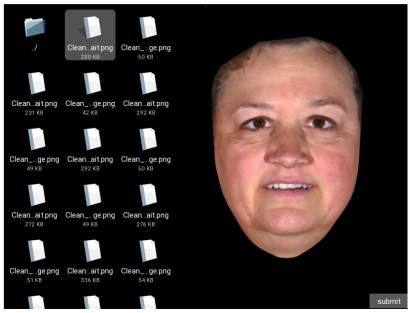


Fig. 8: User Interface

V. Results

First of all we want to know how long we have to train the network.Initially we ran the network for 1-3 epochs and found that output range images had some irregularities.But when we go on increasing the epochs we found that we got an output which was similar to the range image in the dataset.For generating 3-D representation from depth image we first converted depth map generated and original image to point cloud representation.From this representation we generated a wireframe and it was eventually converted to 3-D representation as shown in Figure 9.We have even tried with images other than those in database and we got satisfied results.

VI. Conclusion

In this work, we have implemented a pix2pix GAN model for depth estimation of 2D images for 3D reconstruction.As it is an ill-posed problem many solutions and algorithms have been proposed over decades.One among them is the Pix2pix GAN.It is a method for image-to-image translation using conditional adversarial networks which gives more robust output for depth estimation and 3-D reconstruction.We

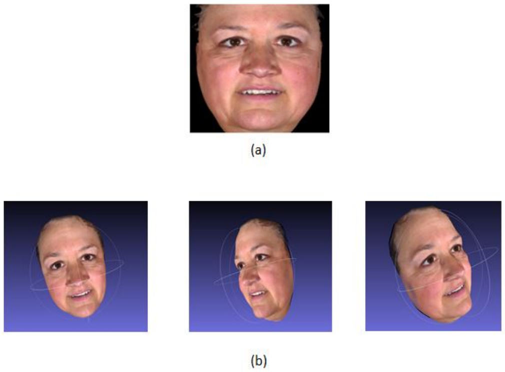


Fig. 9: Figure (a) shows the 2-D image and Figure (b) shows the 3-D representation

have used this model as it is more flexible and adaptable than any other GAN.

Face depth estimation and 3-D reconstruction is an exciting and promising approach,and it can be further extended to medical field where it helps to detect the internal injuries and in insurance field it helps in car damage assessment.

References

[1] R. Zhang, P.-S. Tsai, J. E. Cryer, and M. Shah, “Shape-from-shading: A survey,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 21, no. 8, pp. 690-706, Aug. 1999.

[2] M. J. Brooks and B. K. Horn, “Shape and source from shading,” in Proc. 9th Int. Joint Conf. Artif. Intell., Burlington, MA, USA, Morgan Kaufmann, Aug. 1985, pp. 932-936.

[3] I. Kemelmacher-Shlizerman and R. Basri, “3D face reconstruction from a single image using a single reference face shape,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 2, pp. 394-405, Feb. 2011.

[4] I. Goodfellow et al., “Generative adversarial nets,” in Proc. Adv. Neural Inf. Process. Syst., 2014, pp. 2672-2680.

[5] M. Mirza and S. Osindero. (2014). “Conditional generative adversarial nets.” [Online]. Available: https://arxiv.org/abs/1411.1784

[6] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. (2017). “Image-to-image translation with conditional adversarial networks.” [Online]. Available: https://arxiv.org/abs/1611.07004

[7] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in Proc. Int. Conf. Med. Image Com- put. Comput.-Assist. Intervent. Cham, Switzerland: Springer, Nov. 2015, pp. 234-241.

[8] S. Gupta, K. R. Castleman, M. K. Markey, and A. C. Bovik, “Texas 3D face recognition database,” in Proc. IEEE Southwest Symp. Image Anal. Interpretation (SSIAI), May 2010, pp. 97-100.

[9] J. Zhang, K. Li, Y. Liang, and N. Li, Learning 3D faces from 2D images via stacked contractive autoencoder,” Neurocomputing, vol. 257, pp. 67-78, Sep. 2017.

[10] Tensorflow Documentation, Exponential Decay. Accessed: Nov. 17, 2018. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/train/exponential_decay

[11] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, “Improved techniques for training GANs,” in Proc. Adv. Neural Inf. Process. Syst., 2016, pp. 2234-2242.

[12] H. Fu, Y. Chi and Y. Liang, ”Local Geometry of Cross Entropy Loss in Learning One-Hidden-Layer Neural Networks,” 2019 IEEE International Symposium on Information Theory (ISIT), Paris, France, 2019, pp. 1972-1976.

[13] Hang Zhao, Orazio Gallo, Iuri Frosio and Jan Kautz (2018). ”Loss Functions for Image Restoration with Neural Networks”[Online]. Available://arxiv.org/pdf/1511.08861.pdf

[14] D. P. Kingma and J. Ba. (2014). “Adam: A method for stochastic optimization.” [Online]. Available: https://arxiv.org/abs/1412.6980

[15] YU Y., ZHOU K., XU D., SHI X., BAO H., GUO B., SHUM H.: Mesh editing with Poisson-based gradient field manipulation. TOG (SIGGRAPH ’04) 23 (2004), 641–648.