Computer Vision 2 (2022) **(Due: June 2, 2022)**

Assignment 3 - 2D-to-3D for FaceSwapping

Name: Anne Chel and Venkat Mohit Sornapudi

1 Introduction

This research focuses on face-swapping in both video and images. Accurate face-swapping and face-mimicking techniques have seen a growth in popularity, with extraordinay results in deep fakes. This research first experiments with a DL-based framework in section 3.1. In section 3.2 experiments with modeling the 3D texture of a face from a 2D image using the eigenvectors of a trained PCA model with landmark extraction and optimization are performed, from where camera projection is applied. In section 4 a GAN is implemented to improve the blending of source on target.

2 Methodology

This section discusses all the methods implemented in this research.

2.1 3D-model with PCA

Eigenvectors of the Principal Component Analysis (PCA) describe a certain space. Here, we vary 30 principal components for facial identity (α) and 20 principal components for facial expression (δ) from the mean values of the Morphable model Basel Face Model 2017 (BFM) [1] to build required 3D reconstruction of a given face as an image. This is efficiently done by an iterative Energy optimization method. To implement the optimization method, we first extract the facial landmarks of the given image (using Dlib's face detector and shape predictor) and the 3D reconstructed face that is initialized with zero valued principal components and without any pose transformation (using the given Landmarks68_model2017-1_face12_nomouth.anl indices). Now, thus obtained landmarks are matched (to form pairs) and an error is then computed using sum of mean pixel-wise/pair-wise-euclidean distance and Tikhonov regularization [2] loss (consisting lamda_{α} and lamda_{δ} as weights/parameters). This error is minimized using gradient decent and passing the learned parameters (principal components and pose transformation) to the following iterations. If the error falls down below a set threshold the learning is stopped and the finally obtained parameters are used in the next steps of the pipeline.

2.2 Camera Projection

The appearance of objects change depending on certain views. For instance in 3D games on 2D screens, when you change the view of what your character is seeing, the environment and the objects in it should change appropriately. Similarly, here we want to project the 3D reconstructed face onto the image plane and later lay this planar reconstructed face (mask) over the given facial image. To project the 3D reconstructed face, we first find the corresponding View-port (http://glasnost.itcarlow.ie/~powerk/GeneralGraphicsNotes/projection/viewport_transformation.html) and Projection matrices (https://bit.ly/300gYmf). Then we convert the 3D coordinates into homogeneous coordinate system. Multiplying the homogeneous coordinates with the View-port and Projection matrices we get the required projection. Finally, we convert the result, which will be in homogeneous coordinate system, into normal 3D coordinates.

2.3 Blending with GANs

Generative Adverserial Networks (GANs) are powerful generative models. Where variational auto-encoders aim at modelling a certain latent distribution (often Gaussian) from where new samples can be sampled from, the objective of a GAN is to only generate good samples.
Obtaining the 3D texture of a face and applying the rotation and translation from another face and appropriately render it allows you to face-swap. Often just overlaying these two does not yield natural results, especially in boundary areas, different lightning conditions or different skin types. Therefore, in this research a GAN is

fine-tuned to naturally blend the two components. The architecture of the decoder and generator come from FSGAN[3] and an almost identical loss functions are implemented. Following the approach in [3] the FSGAN receives as input the target picture overlaid with the source face using a mask. This will be concatenated to the original target image and the corresponding mask to create a 7 channel input. The model will not only be updated using the regular discriminator and generator loss, but the generator loss will also consist of the reconstruction loss between the ground truth and the generated image. The ground truth can be composed using several methods. In this research 3 methods are implemented, that is Poisson blending, alpha blending and Laplacian pyramid blending.

3 Results

This section answers all the questions given in the assignment by providing the results of the experiments conducted and analysis/discussion on them.

3.1 (4.1) Deep Learning-based Face Swap

Landmark extraction is task of finding a set of facial landmarks describing interest points spanning the chin, eye, eyebrows, nose and mouth region. Experimentation is performed on a selection of images, varying in lightning conditions, occlusions and pose, visible in figure 1, to investigate the robustness of the landmark detection algorithm from DLib[1] for 68 facial landmarks.

Figure 1: Extreme lightning, pose and occlusion examples

Out of all eight images, only on three images landmarks could be found; these are visible in figure 2. For picture d) in 1 it is not surprising the detection algorithm fails to find a face, as the picture is very dark and the face is hardly recognizable. For e) however the detection also fails, while a clear face is visible may it in extreme lightning conditions. The algorithm detects a face in g) even though almost half of the face is blocked, although in a) the face is partly blocked and here the algorithm fails to recognize a face. In b) a face is recognized although the landmarks are visually off; this could be due to the face mask covering crucial information about the shape of the mouth and nose to form the rest of the facial landmarks.

[1] https://github.com/davisking/dlib

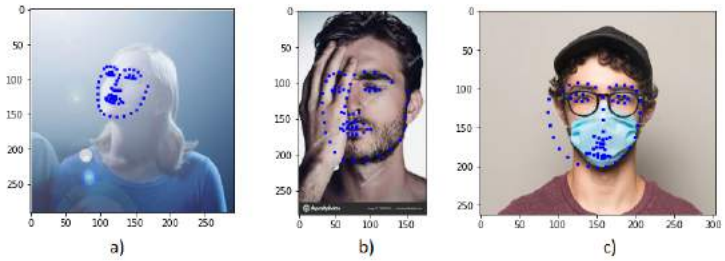

Figure 2: Images from figure 2 were landmarks could be found

Naive faceswapping consists of first finding the interest region of the face spanned with the landmarks, from where this spanned region is overlaid with the target image, see the right component in 3 and 4. This method clearly does not yield visibly pleasing results and fail to migrate the pose and lightning condition from one face to the other. If faces were similar in color, lightning conditions and pose the naive faceswap approach could yield good results, but the aforementioned conditions are always never met.

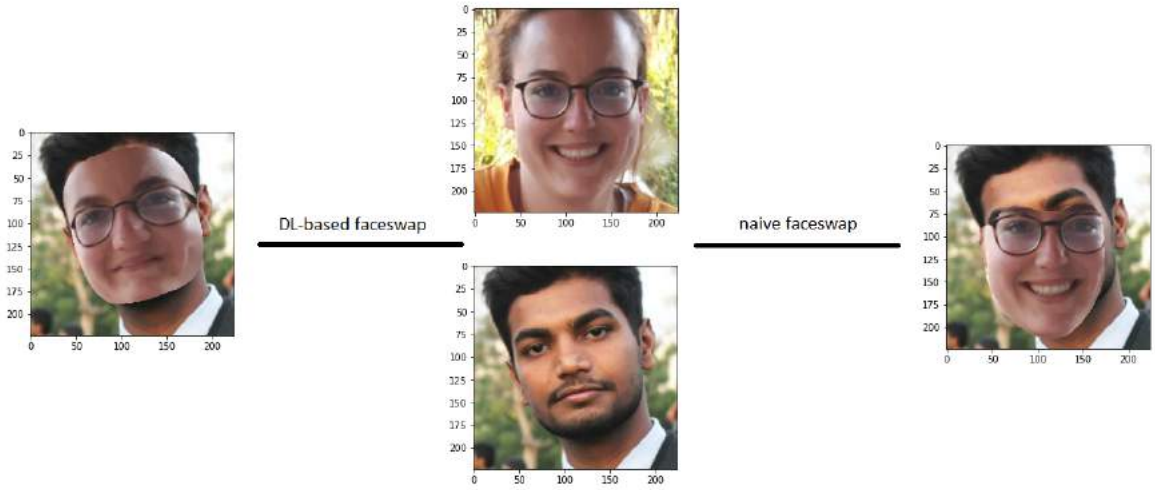

Figure 3: Example faceswap in similar lighting conditions for naive implementation (left) and deep-learning-based faceswap (right).

The deep-learning-based faceswap improves the result from the naive approach. Two examples are again visible in 3 and 4 in the left component. DL-based faceswap is capable to transfer the pose correctly. Although the edges of the spanned face overlaid in the target image are not smooth and this will form a problem in different lightning conditions. This is clearly visible in 4, where although the pose is correctly transformed the result does not look like a real face.

3.2 (4.2) Optimization-based Face Swapping

3.2.1 (4.2.1) Morphable model

In a BFM Model, one generate different facial shapes and expressions using the principal components. A unique face can then be obtained by adding variations to the mean principal components. The shape is controlled by α and the expression by δ. We have uniformly and randomly sampled $\alpha$and δ in the range [-1,1] and built five 3D reconstructed faces as shown in figure 5. We found that when α is increased either in the positive or negative scale, more variation is imposed in the shape of the face. Likewise, when δ is increased in the positive or negative scale, more variation is imposed in the expression.

3.2.2 (4.2.2) Pinhole camera model

Since the BFM face model is represented in 3D coordinates one could directly apply a spatial transformations by using rotation matrix and translation vector. This is demonstrated in figure 6, where a 3D model is rotated

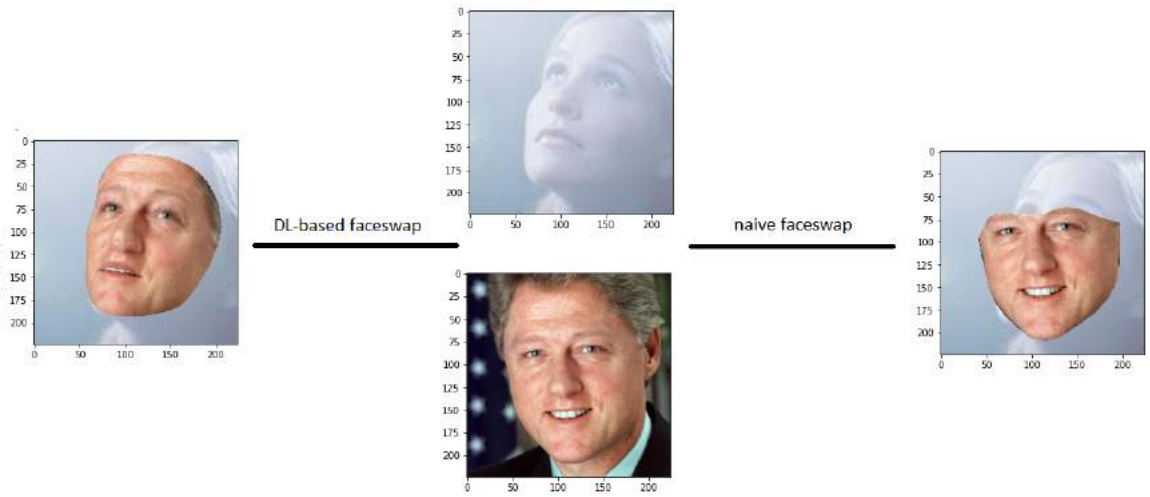

Figure 4: Example faceswap in different lighting conditions for naive implementation (left) and deep-learning-based faceswap (right).

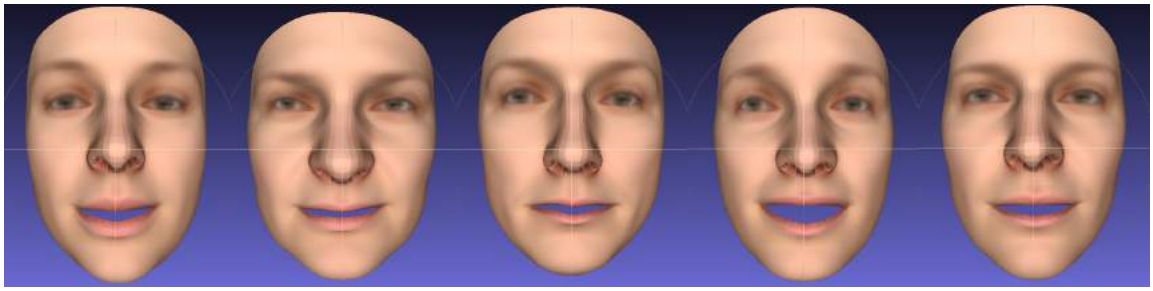

Figure 5: Varied 3D reconstructions obtained using random principal components

by 10 degrees and -10 degrees around the y-axis.

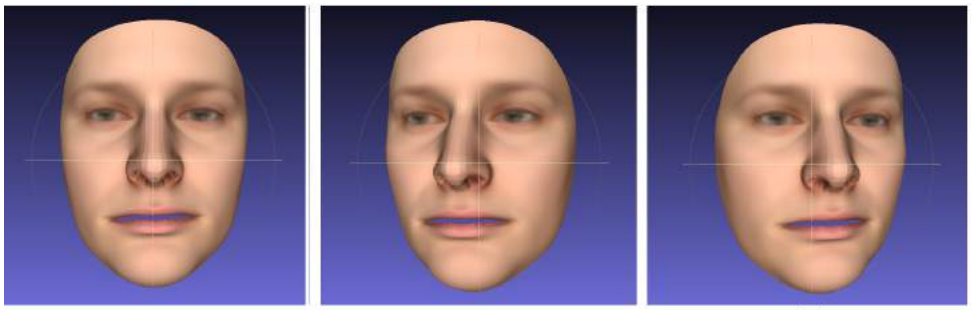

Figure 6: Rotating 3D face: (a) Input 3D face (b) Input rotated by 10 degrees about Oy (c) Input rotated by –10 degrees about Oy

To obtain the corresponding 2D projection of the 3D face, camera projection is applied by means of taking product with respective view-port and perspective projection matrices. To illustrate this, a random 3D face is translated to [0, 0, –500] and rotated by 10 degrees about Oy. The resultant object is projected to 2D camera plane as shown in figure 7 (a) (for the sake of visualization the facial points are scaled appropriately and translated to centre of the image). By using the Landmarks68_model2017-1_face12_nomouth.anl indices the landmarks are identified in the 3D face and marked on the 2D face as shown in 7 (b).

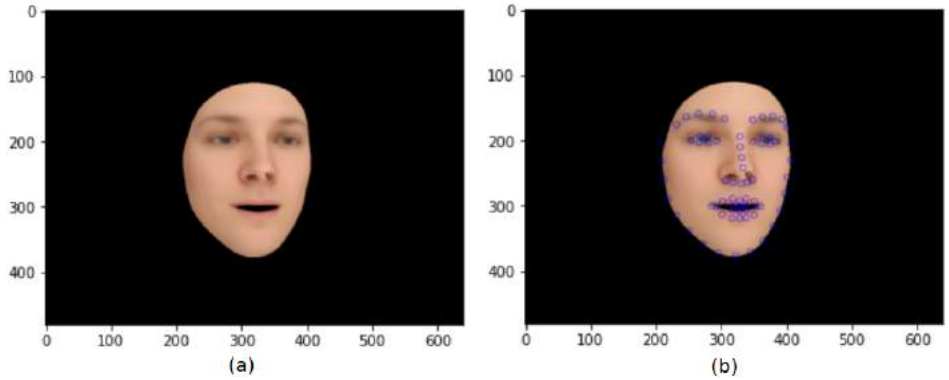

Figure 7: (a) Projected image of the 3D face which is obtained after given translating to [0, 0, –500] and rotating by 10 degrees around Oy (b) Landmarks (marked in blue) of the projected image

3.2.3 (4.2.3) Latent parameters estimation

Here we use the Energy Optimization method to obtain a 3D facial model that fits any given image. In each iteration of the method implemented, a (representing structure) and d (representing expression) are optimized, together with *omega* and t for the pose estimation. After the training/optimization, we obtain a fitted 3D face. Now, this 3d face is then projected onto the camera plane as discussed in the previous subsection. The effectiveness of the optimization can be viewed by comparing the predicted and ground truth landmarks. The predicted landmarks are obtained by using indices from the given file as explained in the previous subsection. The ground truth landmarks are obtained using face detector and landmarks predictor functions from Dlib library. This is clearly visualized in figure 8. We find that the predicted landmarks almost perfectly align with the ground truth landmarks.

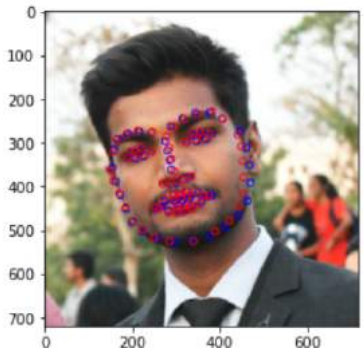

Figure 8: Comparision between detected landmarks (marked in blue) and predicted landmarks (marked in red) of the facial image

In Energy Optimization, the loss function consists a regularization term in which there are 2 weights/parameters namely $lamda_\alpha$ (weight over α square) and $lamda_\delta$ (weight over δ square). Large weights constrain the final principal components to be small. This can be clearly seen from the table 1. Small α values correspond to faces with geometry similar to the mean of the model and small δ values correspond to faces with neutral expressions. So, for instance, if one wants to model faces that have extreme expressions like laughter, sadness etc., the $lamda_\delta$ parameter should be set lower.

Hyperparameters	lamda$_\alpha = 0.8$	lamda$_\alpha = 0.1$
lamda$_\delta = 0.8$	-1.53 | 1.97 | -1.06 | 1.46	-2.49 | 2.08 | -1.09 | 1.51
lamda$_\delta = 0.1$	-1.51 | 1.977 | -1.73 | 2.08	-2.46 | 2.08 | -1.65 | 2.09

Table 1: α_{minimun} | α_{maximum} | δ_{minimun} | δ_{maximum} values obtained for different combinations of hyperparameters

3.2.4 (4.2.4) Texturing

The 2D projection of the fitted 3D face is is with mean color values as shown in figure 9 (a). Overlaying this projection in the reference facial image (input image), as done in figure 9 (b), clearly shows the colors do not match with the reference face. To obtain the right colors for each point, we extract corresponding RGB values from the reference image by bilinear interpolation of neighborhood pixels. By changing the default color (mean color) values of the 2D generated image to the extracted RGB values, we get a image patch which can be used in face swapping (as shown in figure 9 (c)). Overlaying this patch on the reference image 9 (d), we see that the patch exactly fits the image both color-wise and structure-wise. By using these extracted RGB values we can finally get a 3D model of the 2D face from the reference image as shown in 10.

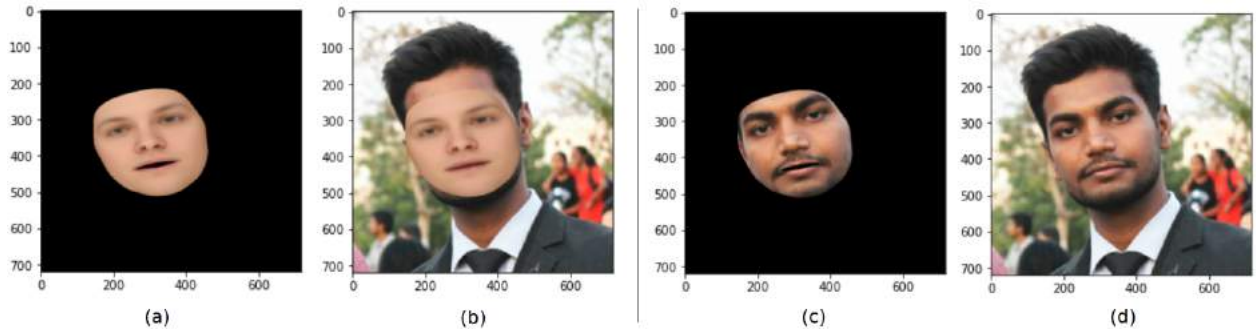

Figure 9: (a) 2D projected face of generated 3D face of mean color values, (b) Image obtained by overlaying 2D face from (a) onto reference image, (c) 2D projected face of generated 3D face with extracted color values, (d) Image obtained by overlaying 2D face from (c) onto reference image

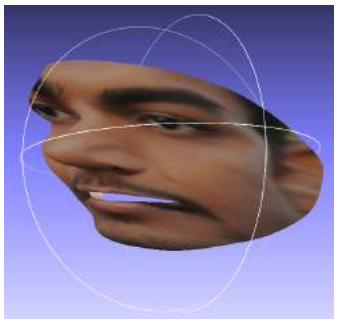

Figure 10: 3D face with extracted RGB colors

3.2.5 (4.2.5) Energy optimization using multiple frames

The pipeline in previous subsection receives one image as input. Here, it is extended such that multiple frames of the same person can be use to get an optimized 3D face of the person.

We tried two ways of using the multiple frames. The first is by taking element-wise mean of α's and δ's, that are obtained by optimizing over the multiple frames, respectively. The result obtained using this method is shown in figure 11. The second is by reusing the trained parameters (α and δ) of previous image as initialized parameters of the successive image in the taken training image sequence. The result obtained using this method is shown in figure 12.

We find very minor differences between the outputs of the 2 methods. This might be partly due to the reason the images used in training were fairly similar (did not change them due to lack of time). The output of first method looks better than the output of the second method. As the final output in the second method is the output from training over final image in the training sequence, the final image should have a greater influence over the final output. We speculate that this should be reason for second method to be performing relatively worse.

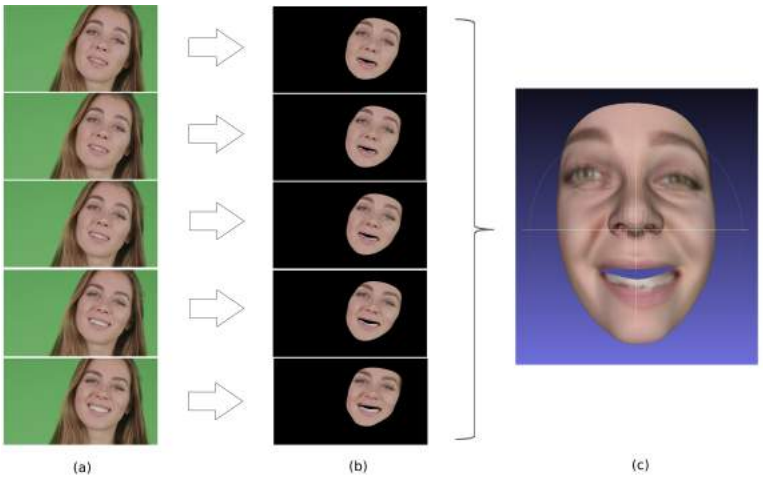

Figure 11: Results obtained by taking mean of trained parameters of the training image sequence. (a) Input frames, (b) Obtained 2D face patches, (c) Final 3D face obtained

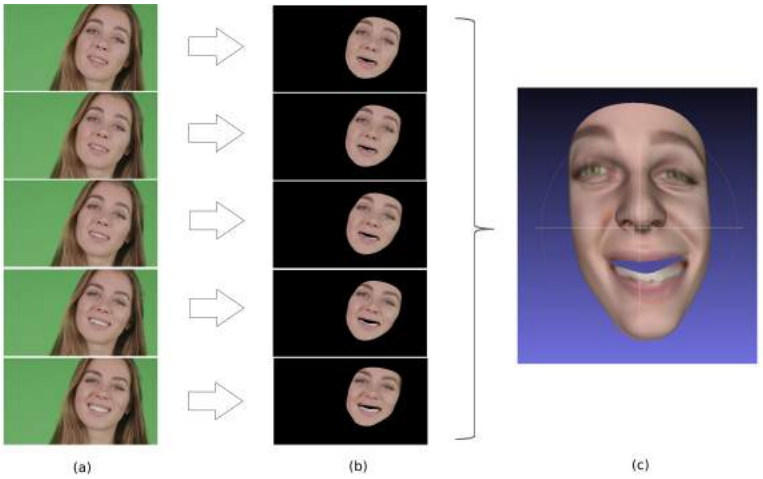

Figure 12: Results obtained by reusing trained parameters of previous image in training over next image of the training image sequence. (a) Input frames, (b) Obtained 2D face patches, (c) Final 3D face obtained

3.2.6 (4.2.6) Face Swapping

In previous subsections we have trained over an image and found α, δ, color values, rotation matrix, and translation vector. Later, we used these α, δ, and color values to build 3D face. Finally, the rotation matrix, and translation vector are used to project the obtained 3D face onto the image plane to get a 2D image patch.

In this subsection follow a similar pipeline to swap faces of 2 persons. First, we train over the images of the persons separately and collect the respective trained parameters. Next, we build 3D model of person's face in the source image using the source image's α, respective δ (if we want final expression of the source image we use source image's δ or else if we want final expression of the target image we use the target image's δ) and source image's color values. Then we rotate and translate thus obtained 3D face using target image's rotation matrix, and translation vector. Later, we check if the centroids and sizes (range between minimum and maximum) of landmarks of target image and the transformed 3D face match. If not, we scale and translate the 3D face accordingly. Finally, we use the x and y coordinates of the 3D face, and source image's color values to get the required image patch. This image patch is overlayed over the target image to produce the face-swapped image.

The results of face swapping using different source, target and expression configurations are displayed in 13. Clearly, there is no much difference between results of using target image's expression and source image's expression; always outputs source image's expression. This has been the case even if we use smaller $lamda_\delta$ weight in regularization loss term. It might be due to the fact that we used small $lamda_\delta$ weights while training on both source and target images. In the next subsection we could see expressions changing in every image; mimicking target image's expression. The only difference is that we used bigger $lamda_\delta$ weight during training over source image and smaller $lamda_\delta$ weight while training over target image. This makes sense, as during

training there is a trade-off between α and δ values and are not independent. So, if we need target image's expression in the final output we should better use smaller $lamda_{\alpha}$ and larger $lamda_{\delta}$ values while training on source image and larger $lamda_{\alpha}$ and smaller $lamda_{\delta}$ values while training on target image.

(Comparision with 4.1 is done in the conclusion.)

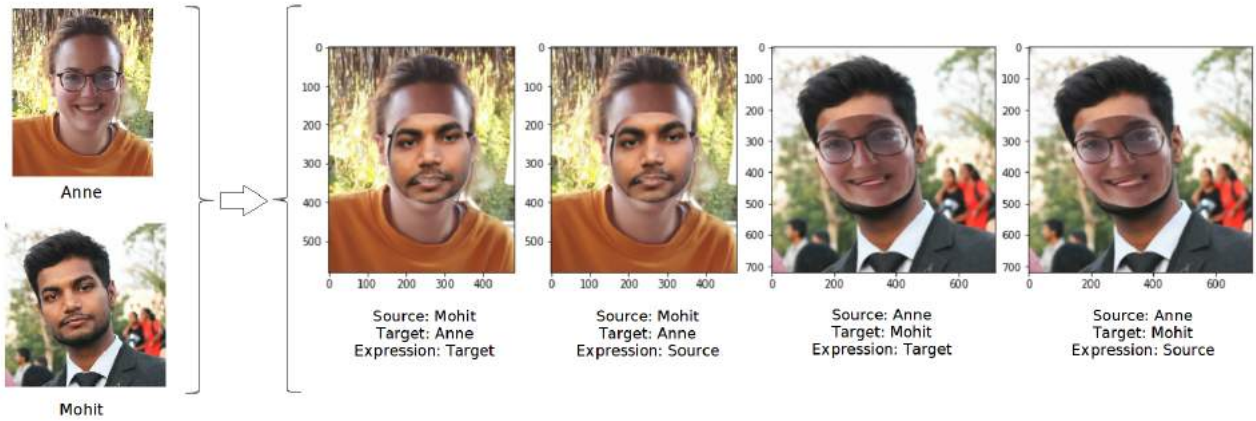

Figure 13: Results of face swapping using different source, target and expression configurations

3.3 (4.3) Face-swapping on Video

Face-swapping on a video is performed. Accesible via: https://drive.google.com/file/d/1R7F_VupOc17WJO_glgFufnYxBU90esH2/view?usp=sharing. The video consisted of a young woman expressing several emotions such that the ability of the model to handle diverse input is tested. Initial results were promising, although the model was slow in handling all frames in the video. Suggesting that a real-time approach is not viable, unless computation time is reduced. Although the expressions and movements are mimicked, the transferred face can look morphed. This could be due to the fact that we stopped training over target images if the loss reaches below 200 (high number), in order to get outputs quickly due to the time constraint we had.

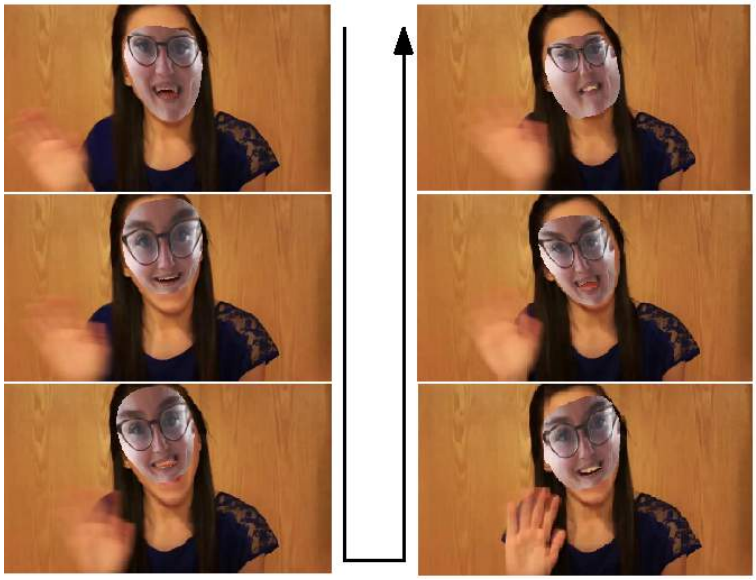

Figure 14: First six sequential frames of the face-swapped video

Introducing a high learning rate in the beginning of training, and reducing this after a certain threshold greatly speeded up the number of iterations and computation time to converge. Initializing alpha and omega with the previous frame also reduced computation time, since consecutively frames do not really change much in both texture and geometry. This also improves the quality of the obtained 3D model, since we will definitely not be stuck a local minima if the first parameter variables are obtained correctly.

3.4 (4.4) Blending

The data-set class is implemented in the provided framework. Here the full data-set is loaded and returns the target, source, swap and mask. In addition the ground truth and transferred input to the generator also created here and passed to the data-loader. This transferred input is obtained by overlaying the swapped image (i.e. the face of the source aligned to the pose of the target) with the target image using the provided mask. Different blending functions for the ground truth can be selected. Besides Poisson blending (this is the blending used in all ground truth visualizations in this section in 161817), alpha blending and Laplacian Pyramid blending are implemented. The latter two are visible in figure 15. From initial experimentation Poisson gives smoother blended outputs than alpha- and Laplacian Pyramid blending.

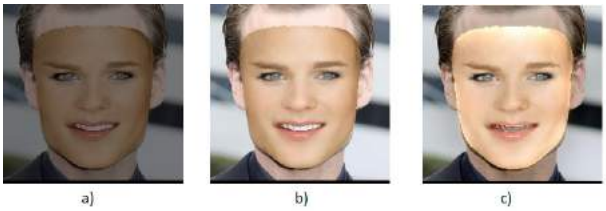

Figure 15: Different groundtruth blending examples a) Alpha blending with alpha=0.5 combining swap and target*(1-mask)) b) Alpha blending with alpha=0.8 for the swap+(1-mask) image and 1-alpha for the target image c) Laplacian pyramid blending for 6 pyramids

The GAN losses are implemented. The loss for the discriminator consists of the generator loss and reconstruction loss. For the generator loss the MSE between the output of the discriminator and a tensor of same shape full of 1's is taken. These 1's represent that the generator wants to fool the discriminator such that the discriminator thinks the generated sample is a true picture of a face. The reconstruction loss consists of the provided VGG loss and the provided pixel Loss. Where the first focuses on high frequency details and the latter on correct color formation. The total GAN loss is defined as weighted combination of the generator loss (0.001) and the reconstruction loss (pixel loss is weighted with 0.1 and VGG loss weighted with 1).

The losses of the discriminator consists of a weighted combination (0.5) of the MSE on the output of the discriminator for fake, i.e. the generated sample from the generator (compare with 0's) and real (compare with 1's) images.

To analyze the effect of the discriminator the network is trained without the discriminator. Now the losses only comprise the reconstruction loss back-propagating to the generator. The hypothesis is that if the discriminator is removed the generator will simply mimic the ground truth blending function. Example of a sample generated by the generator together with the source, target, swapped and ground truth blended function using Poisson blending is visible in figure 16. Here we trained for 2 full epochs on the whole data-set with the aforementioned settings.

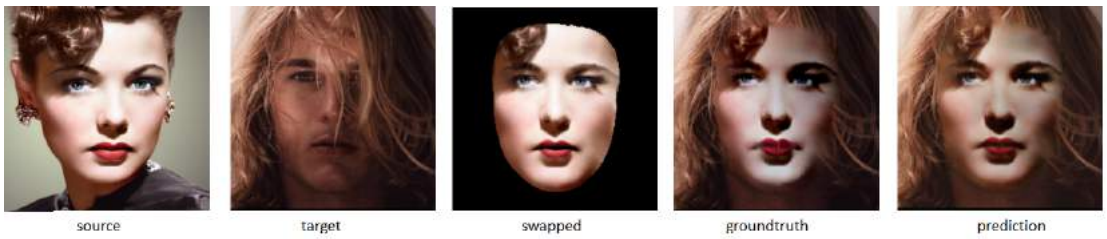

Figure 16: Results training without discriminator

Often the Poisson blending function provides good results as is for instance visible in the previous figure 16. The blending function however fails to produce 'real' faces when faces are occluded, the transferred pose in the swapped image is vague or incorrect, the border of the mask does not blend well, for instance with different hairlines, or some expressive color dominates the blending. Examples of this are visible in figure 17.
If one want to generate new images, one could train a GAN purely with GAN losses. However this poses no real restrictions on the generated samples, as long as they are fooling the discrminator. In faceswapping the sample space is constraint, where the generated sample should contain the exact facial identity regions, i.e. the source face must still be recognizable in the generated sample while also looking 'real'. Therefore the GAN consists of the generator loss (i.e. the success of fooling the discriminator) and the previously mentioned reconstruction

Figure 17: Failing examples of obtained groundtruth via Poisson Blending with a) occlusion b) dominated color dominating the face color c) mismatch edges d) incorrect obtained pose transformation

loss. Such that the generated samples are constrained to the specific swapped image and target image and the generator can not just generate a 'real' face that looks nothing like target or source.

The GAN with Poisson blending groundtruth is trained for 7 epochs. In the final epoch the discriminator loss became much lower than the generator loss, therefore the discriminator was frozen for the last epoch to only train the generator.

The resulting image predictions from the generator show that the GAN still resembles the Poison distribution. The hope was that using a GAN, problems such as present in figure 17, would be resolved by the generator capability to make new samples. This trend is however not significantly present. Suggesting that the GAN should continue training or the generator weighing scale should be set higher/reconstruction loss weighted lower. There is a trade-off between producing samples close to the groundtruth and the freedom of the GAN to produce new samples. There are cases where samples produced by GAN feel more 'real', these are visible in figure 18, where the ground truth for both a and b yields inappropiate shading and in c the Poisson blending makes the red lips continue on the teeth.

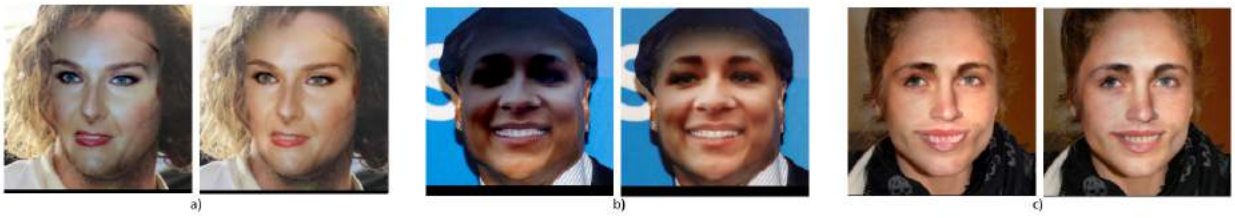

Figure 18: Example predictions of GAN (right in each pair) and groundtruth obtained with Poisson blending (left in each pair)

To perform quantitative analysis the FIP scorehttps://github.com/mseitzer/pytorch-fid for a small subset is calculated between the true image distributions (targets) for 1) the groundtruth images obtained with Poisson blending and 2) the predicted images of the implemented GAN. The FIP score for 1) is 42.59 and the FIP score for 2) is 45.99. Although the scores are close, Poisson Blending produced more natural images than the trained GAN. Suggesting that the GAN should definitely be more trained and tweakes to bring the FIP score down and produce 'realer' samples.

4 Conclusion

In this assignment face-swapping is analyzed through different methods. The naive approach is to simply extract the facial regions via landmarks and transfer this region to the target. If faces were similar in color, lightning conditions and pose the naive face-swap approach could yield good results, but the aforementioned conditions are always never met. Experimentation is performed with obtaining 3D structure and expression via the BFM, from where the object can be projected into the 2D image rendered with the correct RGB vertices. Face-swapping is performed on a video and on images. Results show pose can be correctly captured, although sometimes somewhat morphed, but the blending is not optimal and a clear boundary region is visible between the swapped region. The DL-approach would therefore be more appriopiate for video face-swapping. To overcome this problem a GAN is implemented. Unfortunately using GAN did not improve over the groundtruth Poisson blending result, and the trained GAN needs to be further trained to reduce the FIP score. The trained GAN however can produce beautiful results and the blending, although similar to Poisson looks more 'real' than using no blending at all as in other approaches.

5 Self-Evaluation

Anne Chel and Venkat Mohit Sornapudi worked equally on code and report.

References

[1] Thomas Gerig et al. *Morphable Face Models - An Open Framework*. 2017. DOI: 10.48550/ARXIV.1709.08398. URL: https://arxiv.org/abs/1709.08398.

[2] Gene H. Golub, Per Christian Hansen, and Dianne P. O'Leary. "Tikhonov Regularization and Total Least Squares". In: *SIAM J. Matrix Anal. Appl.* 21.1 (Oct. 1999), pp. 185–194. ISSN: 0895-4798. DOI: 10.1137/S0895479897326432. URL: https://doi-org.vu-nl.idm.oclc.org/10.1137/S0895479897326432.

[3] Yuval Nirkin, Yosi Keller, and Tal Hassner. "FSGAN: Subject Agnostic Face Swapping and Reenactment". In: *CoRR* abs/1908.05932 (2019). arXiv: 1908.05932. URL: http://arxiv.org/abs/1908.05932.