Neural Emotion Director: Speech-preserving semantic control of facial expressions in “in-the-wild” videos (Supplementary Material)

Foivos Paraperas Papantoniou[1] Panagiotis P. Filntisis[1] Petros Maragos[1] Anastasios Roussos[2,3]

[1]School of Electrical & Computer Engineering, National Technical University of Athens, Greece

[2]Institute of Computer Science, Foundation for Research & Technology - Hellas (FORTH), Greece

[3]College of Engineering, Mathematics and Physical Sciences, University of Exeter, UK

A. Rendering of NMFC and Detailed Shape Images

As mentioned in Sec. 3.3 of the main paper, we follow [5] and map the manipulated 3D face geometry to a convenient representation for our *Neural Face Renderer*. In more detail, we render an RGB image of the 3D face mesh under the manipulated expressions and pose of each given frame using the following semantically-meaningful color-coding scheme: Every vertex of the 3D mesh triangulation of the adopted FLAME face model [9] is uniquely colored with RGB values that are directly mapped from the XYZ coordinates of the mean face's 3D mesh, after normalizing them to $[0, 1]$, see *e.g.* 3^{rd} row of Fig. 1. As in [5], we use the term *NMFC (Normalized Mean Face Coordinate)* images for these rendered RGB images (i.e. $\mathbf{NMFC} \in \mathbb{R}^{256\times256\times3}$).

As also mentioned in the main paper, we use the so-called **detailed shape images** as additional conditional input for our *Neural Face Renderer*. In more detail, DECA [6] estimates not only a standard 3D face reconstruction but also a person-specific detail vector $\boldsymbol{\delta} \in \mathbb{R}^{128}$ for each frame, which improves upon previous methods by adding mid-frequency details to the face geometry through a UV displacement map. We take advantage of this and deviate from [5] (which is based on older 3D face modelling approaches) by adding the resultant detailed shape images $\mathbf{S}$ as an extra conditional input for the face renderer. To be more precise, the detail vector $\boldsymbol{\delta}$ is used in combination with the manipulated expressions to generate detailed 3D face shapes, which we then render (using conventional 3D graphics as in the case of NMFCs) to create the so-called detailed shape images $\mathbf{S} \in \mathbb{R}^{256\times256\times3}$, see *e.g.* 4^{th} row of Fig. 1. In contrast to NMFCs, for this rendering we use constant gray colouring of the 3D mesh and standard smooth shading, since we want fine geometric details (wrinkles, dimples, etc.) to be clearly visible and help in the conditioning of the neural renderer, in terns of photo-realism of the synthesized face images.

An illustration example of all types of conditional inputs that we feed our *Neural Face Renderer* with is provided in Fig. 1.

B. Face alignment

In this section, we provide more details about face alignment, which is one of the main steps in the *3D Face Analysis* module of our pipeline (see Sec. 3.1 of the main paper). Face alignment is an essential step in most **face-swapping** methods (*e.g.* [12]), where combining footage from 2 different actors in totally different poses requires their faces being brought in correspondence. Although our main aim is to generate the target actor under similar conditions to those seen during training, we found that face alignment is still useful, as it boosts our Neural Face Renderer's generalization ability. In particular, we observed that the renderer struggles to produce novel expressions if those are not present in the training footage under the same pose and position. Therefore, at each given frame, we use Procrustes analysis to estimate a 2D similarity transformation matrix between the 68 extracted landmarks and the corresponding landmarks of a mean face template. The masked input face images, as well as the NMFC, detailed shape, and eye-gaze images are then warped according to this transformation, see *e.g.* last 4 rows of Fig. 1. Furthermore, to avoid jittering artifacts in the aligned images we follow [11] and average the landmarks extracted from multiple slightly displaced versions of the original face image.

C. Blending

Here, we provide more details about Blending, the final step of our *Photo-realistic Synthesis “in the wild”* module (see Sec. 3.3 of the main paper). The seamless composition of the generated face onto the original scene is

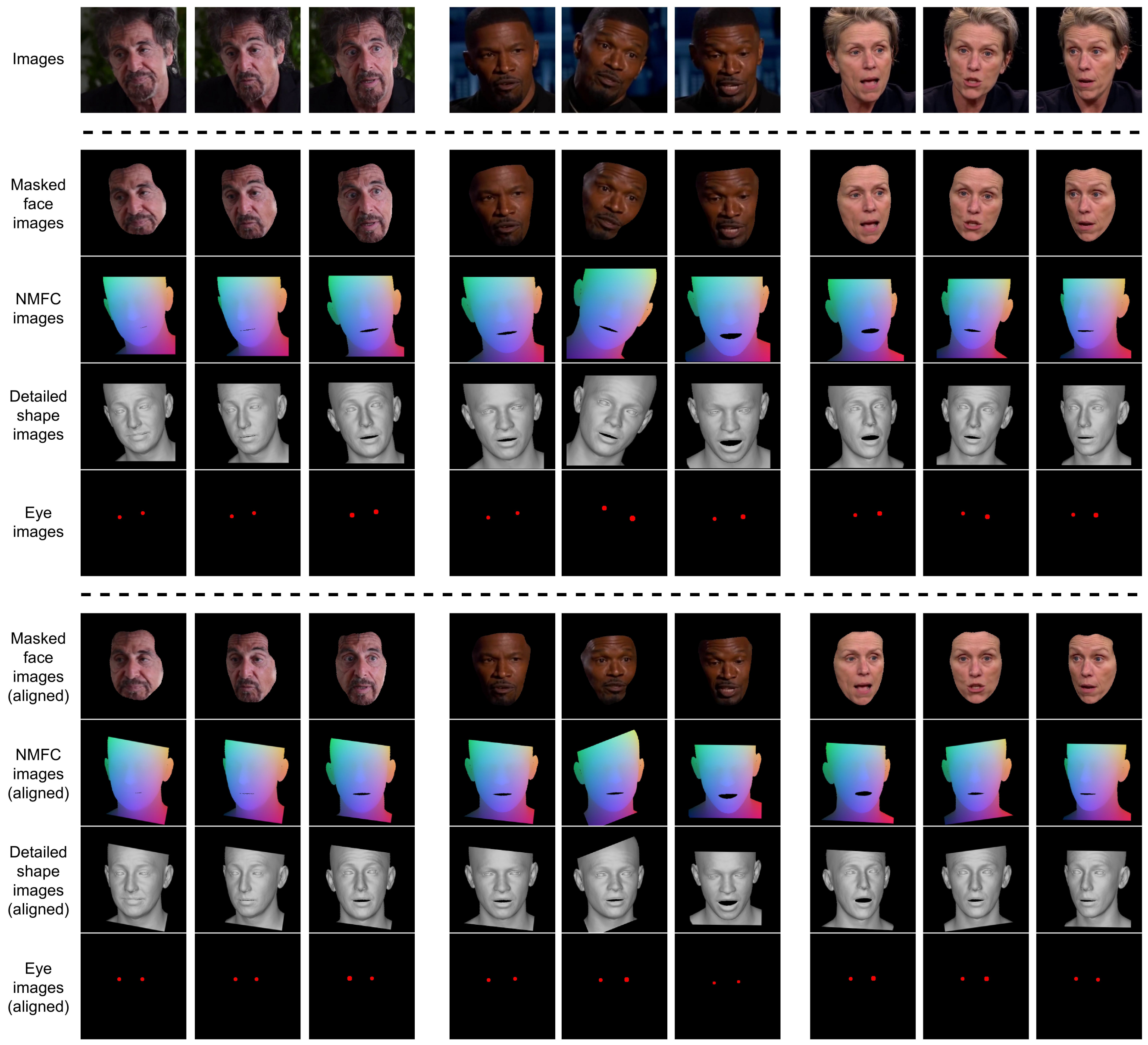


Figure 1. Example frames of cropped face images (1st row), their masking after face segmentation (2nd row) as well as their corresponding NMFC, Detailed Shape and Eye images (next 3 rows). The last 4 rows visualize the warped versions of the Masked face, NMFC, Detailed Shape and Eye images, as a result of the face alignment that we apply. Note that the last 3 rows (aligned versions of NMFC, Detailed Shape and Eye images) correspond to the conditional inputs for our *Neural Face Renderer*. Note also that in this visualization, no emotion manipulation has been applied, which corresponds to the self-reenactment scenario used during training of our *Neural Face Renderer*.

achieved through multi-band blending [1]. In particular, we construct the Laplacian pyramids of both images and perform blending on each level of the pyramid independently using a softly eroded version of the face mask. The blended image is then obtained by reconstructing it from the final pyramid and is placed in the exact same position of the full frame from where it was cropped, thus, fully reproducing the original video, independently of its spatial resolution.

D. Training of the Emotion Manipulator: Loss Functions

In Sec. 3.2.1 of the main paper, we briefly describe the adopted loss functions for training our *Emotion Manipulator*. Here, we expand the presentation providing more details and the relevant mathematical formulas:

The networks of the *Emotion Manipulator* (Translator G, Style Encoder E, Mapping network M and Discriminator

D) are updated based on the following loss functions:

• **Adversarial loss:** We use LSGAN [10] with labels b=c=1 for real samples and label a=0 for fake ones, resulting in the following adversarial objectives for the Discriminator D:

$$\mathcal{L}^{D}_{adv} = \frac{1}{2}\mathbb{E}_{\mathbf{s},y}[(D_y(\mathbf{s})-1)^2]+\frac{1}{2}\mathbb{E}_{\mathbf{s},\tilde{y},\tilde{\mathbf{d}}}[D_{\tilde{y}}(G(\mathbf{s},\tilde{\mathbf{d}}))^2] \quad (1)$$

... and the Translator G:

$$\mathcal{L}^{G}_{adv} = \frac{1}{2}\mathbb{E}_{\mathbf{s},\tilde{y},\tilde{\mathbf{d}}}[(D_{\tilde{y}}(G(\mathbf{s},\tilde{\mathbf{d}}))-1)^2] \quad (2)$$

This way the mapping network M learns to output the speaking styles that belong to the emotional domain $\tilde{y}$ and the translator to produce sequences of the target domain that are indistinguishable from the real ones.

• **Style reconstruction loss:** As in [3], we make sure the output sequence reflects the given style by using a loss that enforces the style vector of the translated sequence, as extracted by the style encoder E, to match the desired one:

$$\mathcal{L}_{sty} = \mathbb{E}_{\mathbf{s},\tilde{\mathbf{d}}}[||\tilde{\mathbf{d}} - E(G(\mathbf{s},\tilde{\mathbf{d}}))||_1] \quad (3)$$

• **Cycle consistency loss:** We use the cycle consistency loss [2, 16], which encourages the translator to produce sequences that preserve the content of the input sequence, so that the input sequence can be reconstructed by translating the output sequence back to the original style $\hat{\mathbf{d}} = E(\mathbf{s})$, as extracted by E:

$$\mathcal{L}_{cyc} = \mathbb{E}_{\mathbf{s},\tilde{\mathbf{d}}}[||\mathbf{s} - G(G(\mathbf{s},\tilde{\mathbf{d}}),\hat{\mathbf{d}})||_1]\,. \quad (4)$$

• **Speech-preserving loss:** As observed in [7], the cycle consistency loss does not always guarantee that the original mouth motion related to speech is preserved by the translator. To this end, we take advantage of the adopted FLAME face model [9], which explicitly controls the jaw opening through the 1^{st} jaw articulation parameter. Thus, we add an extra constraint to the total objective, that takes into account only this mouth-related parameter, instead of the whole expression vector as in [7]. We note here that we seek to faithfully alter the conveyed emotion of a sequence and this usually requires increasing or decreasing the mouth opening, *e.g.* to show anger or the neutral emotion respectively. Hence, the jaw opening of the original and the translated sequence have to be highly correlated, but not identical. This leads us to define our speech-preserving loss in terms of the **Pearson Correlation Coefficient (PCC)** between the original and the translated jaw variable within a sequence:

$$\mathcal{L}_{mouth} = -\mathbb{E}_{\mathbf{s},\tilde{\mathbf{d}}}[\rho_{\boldsymbol{\epsilon}_1,\tilde{\boldsymbol{\epsilon}_1}} + \rho_{\tilde{\boldsymbol{\epsilon}}_1,\hat{\boldsymbol{\epsilon}_1}}] \quad (5)$$

where $\boldsymbol{\epsilon}_1 \in \mathbb{R}$ denotes the first component of the expression vector (jaw opening) and $\rho_{X,Y}$ is the PCC between two variables X, Y. The negative sign originates from the fact that, ideally, we would like the PCC to be maximized, whereas the mouth loss to be minimized. The objective is calculated in a symmetric way, where $\mathbf{s}$ is the original sequence, $\tilde{\mathbf{s}} = G(\mathbf{s},\tilde{\mathbf{d}})$ the translated and $\hat{\mathbf{s}} = G(\tilde{\mathbf{s}},\hat{\mathbf{d}})$ the reconstructed one. All statistical values are calculated as arithmetic means within the N occurences of a sequence. By maximizing the above loss, we manage to balance our challenging and contradictive goal of altering the emotion without distorting the perceived speech (see Fig. 3 of the main paper).

• **Overall objectives:** The objective for G, E and M that is minimized during training is the following:

$$\mathcal{L}^{G,E,M} = \mathcal{L}^{G}_{adv} + \lambda_{sty}\,\mathcal{L}_{sty} + \lambda_{cyc}\,\mathcal{L}_{cyc} + \lambda_{mouth}\,\mathcal{L}_{mouth}$$

whereas the objective for D is: $\mathcal{L}^{D}$=$\mathcal{L}^{D}_{adv}$.

It is worth mentioning that we do not use a diversity sensitive loss as in the original StarGAN v2 [3], since we found it to be not necessary. Also, for balancing our objectives we use $\lambda_{cyc} = \lambda_{sty} = \lambda_{mouth} = 1$, since we have experimentally observed that this choice yields high-quality results.

The *Emotion Manipulator* is trained once (see Fig. 2 (a)) and can then be run on the fly for altering the expressions of every new actor.

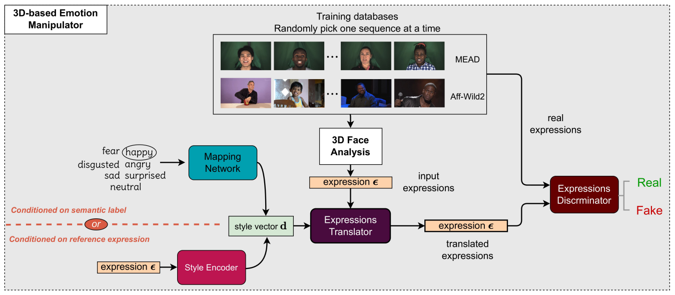
E. Datasets for Experiments

In this section, we provide more details about the two datasets used in our experiments:

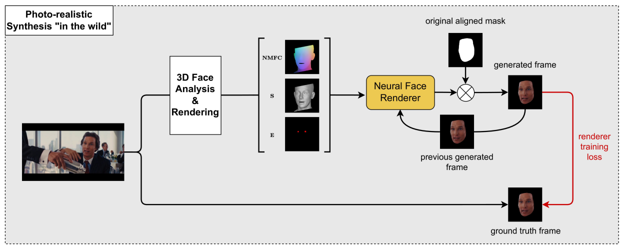
YouTube Actors dataset: We collected a small dataset from 6 YouTube videos (having Creative Commons license) that included facial videos of 6 celebrity actors during film scenes, TV shows and interviews under "in-the-wild" conditions. Short video clips with duration from 2 to 7 mins that capture the actors during talking and performing were excerpted from these YouTube videos and constituted our dataset. The videos of our YouTube dataset were at 30 fps with a spatial resolution of 1280×720 pixels.

MEAD dataset: We chose 3 actors from the recent MEAD database [15]. These actors were not included in the training set that we used for our *Emotion Manipulator*. For every actor, we selected 30 videos for each of the 6 considered emotions (happy, angry, surprised, fear, sad, disgusted) plus neutral, resulting to a total of 630 videos from MEAD. We note that while this database provides emotional videos in 3 different intensity levels, we use only the ones with the highest intensity per emotion. These videos are at 30 fps, 1920×1080 pixels and have an average duration of 4 secs.

Please note that, in both datasets, the selection of the specific actors was done taking into account gender and ethnic group variability. Also, roughly 10% of each actor's footage was kept as test data for the experiments (and 90% as train data for the neural face renderer).



(a) Multi-person training



(b) Person-specific training (fine-tuning)

Figure 2. The 2 trainable components of our *Neural Emotion Director (NED)* are trained separately. **(a)** The *Emotion Manipulator* is trained on person-agnostic expression data, extracted from 2 large video databases with emotion annotations (Aff-Wild2, MEAD). This trained *Manipulator* can then be used for translating the expressions of any new given actor. **(b)** The *Neural Face Renderer* is trained independently for each new actor, by fine-tuning the pre-trained meta-renderer on the training footage of the given actor in a self-reenactment fashion.

F. Training of the Neural Face Renderer: Meta-renderer

Our proposed method for increasing the expressive variability of faces generated by our *Emotional Face Renderer* for actors found in YouTube videos, uses the following scheme:

• First, we train a single **meta-renderer** on a collection of videos consisting of our **YouTube Actors dataset** (extended with 2 more YouTube actors for greater variability) as well as an extension of our selection from the **MEAD dataset** that we make for our experiments (made of 2 more MEAD actors, i.e. 5 in total). In this stage, the meta-renderer struggles to disentagle the different actors but learns to transfuse the expressivity of the MEAD actors to the YouTube actors.

• Then we fine-tune our **meta-renderer** independently for each actor for a few epochs. This step resolves the "identity confusion" caused by the previous stage, without over-adapting to the given actor's expressions.

We found that fine-tuning the **meta-renderer** on a new actor that was not used in the the first stage has similar effects. This is especially important because it means that one does not have to repeat the lengthy pre-training each time he/she wants to apply the method to a new actor. **In other words, for manipulating a new video with a never-before-seen face, the only model that needs to be trained is a new person-specific neural face renderer (generator and discriminator) for synthesizing the new face**. For this, we can simply use the input video as training footage to fine-tune the pre-trained meta-renderer (see Fig. 2 (b)). Nevertheless, performance slighly increases when the actor's footage is used in the training of the **meta-renderer**. Therefore, for obtaining the results reported in the main paper all our actors were included in the training of the **meta-renderer**. Overall, for multi-person pre-training (meta-renderer) we used ∼200K frames (60% from MEAD, 40% from YouTube), while for person-specific fine-tuning we typically use ∼ 5-13K frames.

G. Additional Qualitative Results

We provide additional qualitative results of our method in the form of static frames for the actors of our YouTube dataset in Fig. 3. Our method translates the expressions of the actors to any of the 6 basic emotions plus neutral or to any given reference style, regardless of the emotion of the input video. We observe that we achieve highly-realistic results.

In addition, Fig. 4 visualizes some random cases from the quantitative comparison of Sec. 4.1 of the main paper. In particular, we visualize the pixel distance between the "self-translated" frame (i.e. translated to the same emotion that it is labelled as) and ground truth frame as a heat map in the face area. As also reported in the main paper, we observe that our method performs better in preserving the original emotion without distorting the characteristics of the specific identity, which results in lower FAPD values than all the other methods.

H. Detailed scores for the user study on MEAD database

As mentioned in Sec. 4.2 of the main paper, our second user study asked participants to both evaluate the realism and recognize the emotion of the videos shown to them (on MEAD actors). In Tab. 1 we provide a more extended presentation of the relevant results, by providing detailed realism scores for each of the five points of the adopted Likert scale. As can be seen, videos generated with our method were most frequently rated with 3, while the other methods achieved 1 or 2 as the most frequent rating. This holds true even for each basic emotion individually. In particular, *surprised* is the only category where our most preferred rating dropped to 2, whereas for the other methods, only 2

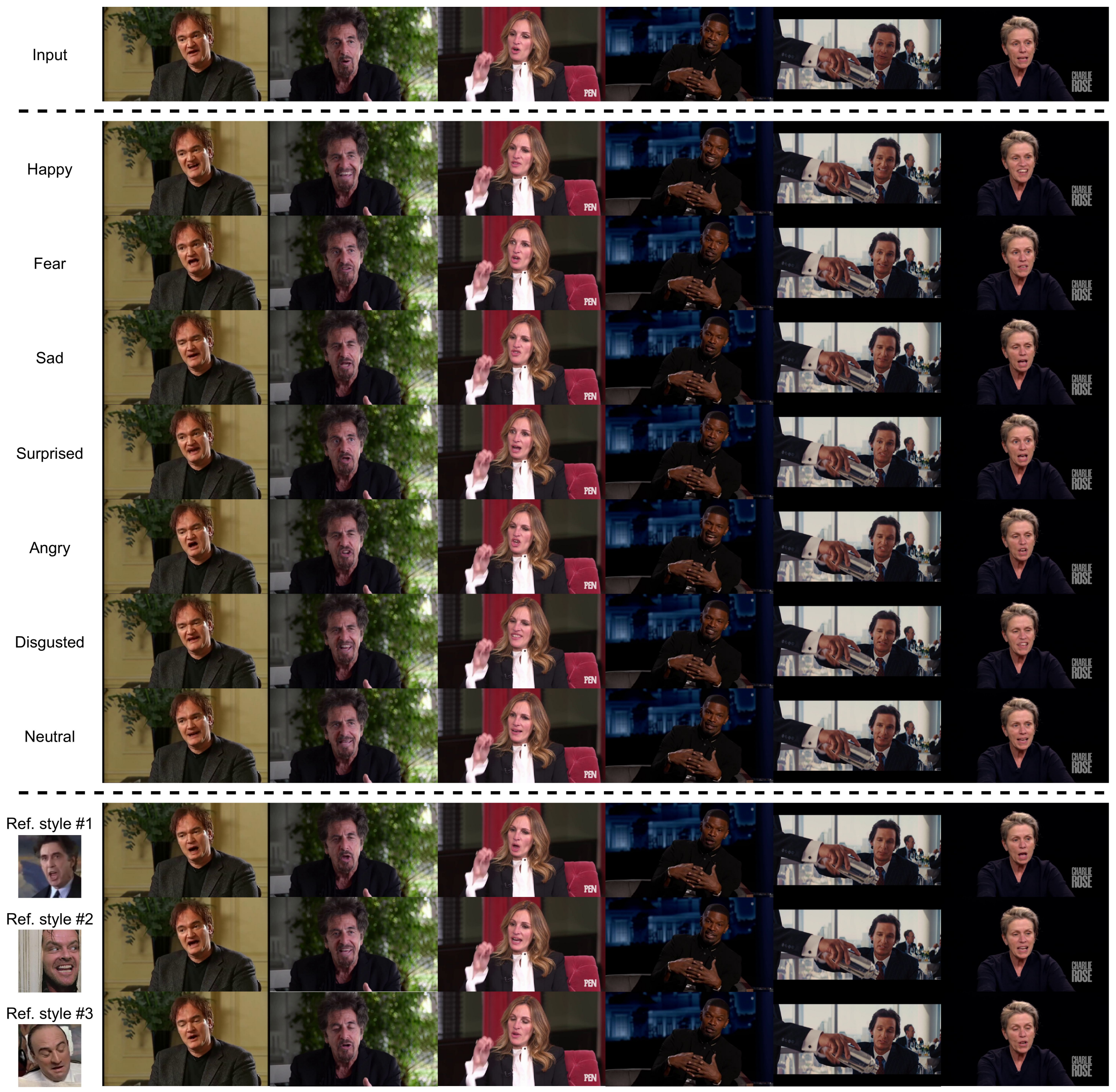


Figure 3. Additional visualizations of our results on the YouTube Actors dataset for random frames of the videos. The first row shows the real frames. The next seven rows correspond to expression manipulation based on categorical labels, while the last three rows imitate the expressive style of three exemplar reference clips. Please zoom in for details.

of the considered categories yielded a most frequent rating higher than 1 (*sad* and *disgusted* for GANmut [4], *fear* and *surprised* for DSM [13]).

Finally, we further elaborate on the emotion recognition results of Tab. 1 through confusion matrices in Fig. 5. As can be seen, *sad* and *surprised* are mostly recognized as *angry* for our method, while the misclassifications of the DSM results seem more mixed and unclear. Moreover, users struggle to separate *fear* and *surprised* in real videos, whereas this is successfully achieved for GANmut, confirming that GANmut emotions seem more 'stereotypical' than the real ones produced by the actors.

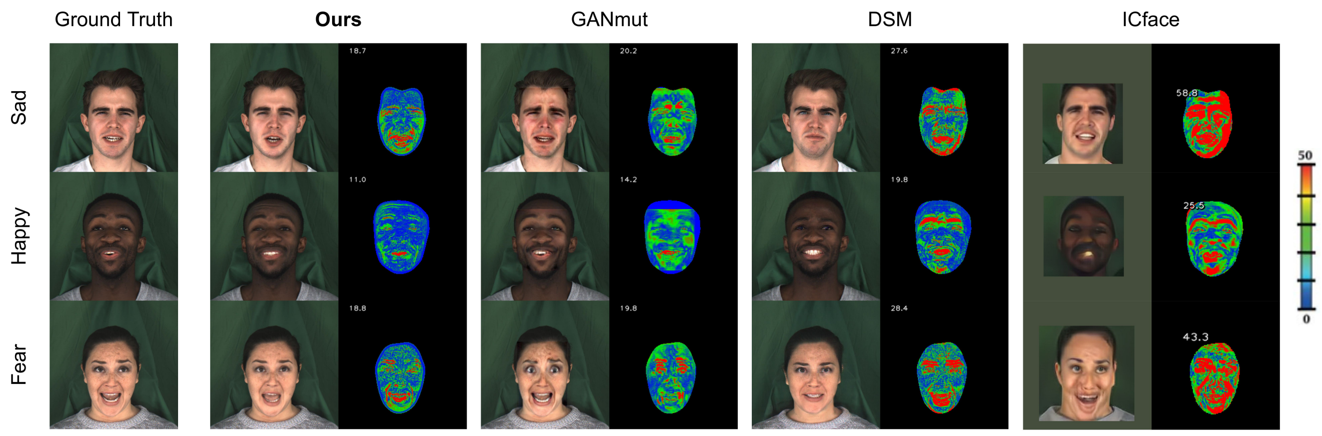
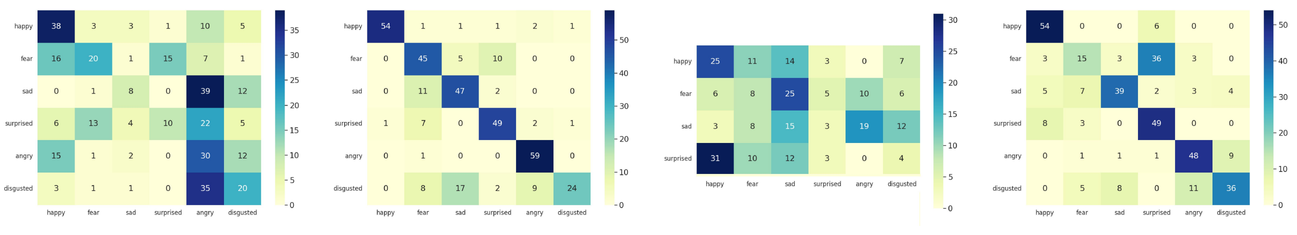


Figure 4. Quantitative comparison with state-of-the-art methods in the emotional "self-translation" experiment on the MEAD actors. For each method, the left image is the generated one and the right image is the error between the generated and the input image, visualized as a heat map within the face mask. White numbers inside the heat maps denote the average error for the given frame (Face Average Pixel Distance - FAPD). We observe that, in all cases, our method yields the lowest average errors. Note that the considered range of pixel values is [0,255].



Ours GANmut DSM Ground Truth

Figure 5. Per-method confusion matrices for the classification of emotions regarding the user study on MEAD. Row labelling corresponds to the ground truth annotations, while column labelling to the predicted ones.

	Realism																								Accuracy			
	Ours						GANmut						DSM						Ground Truth						Ours	GANmut	DSM	Ground Truth
	1	2	3	4	5	real	1	2	3	4	5	real	1	2	3	4	5	real	1	2	3	4	5	real				
happy	15	17	**18**	10	0	17%	**39**	8	11	1	1	3%	**26**	23	6	2	3	8%	0	1	11	15	**33**	80%	63%	90%	42%	90%
fear	5	10	**26**	15	4	32%	**34**	14	8	2	2	7%	19	**27**	8	5	1	10%	2	4	14	20	**20**	67%	33%	75%	13%	25%
sad	4	15	**23**	15	3	30%	9	**22**	18	7	4	18%	**20**	19	14	7	0	12%	2	11	14	15	**18**	55%	13%	78%	25%	65%
surprised	7	**21**	19	12	1	22%	**35**	13	7	4	1	8%	19	**25**	12	3	1	7%	0	1	10	20	**29**	82%	17%	82%	5%	82%
angry	3	18	**24**	9	6	25%	**38**	11	5	3	3	10%	-	-	-	-	-	-	0	2	11	**29**	18	78%	50%	98%	-	80%
disgusted	1	17	**18**	20	4	40%	12	**25**	11	11	1	20%	-	-	-	-	-	-	2	3	15	**21**	19	67%	33%	40%	-	60%
avg.	6	16	**21**	14	4	28%	**28**	16	10	5	2	11%	21	**24**	10	4	2	9%	2	4	12	20	**23**	71%	35%	77%	21%	67%

Table 1. Detailed realism ratings and emotion classification accuracy of the user study on MEAD. Columns 1-5 show the number of times that users gave this realism rating. The column "real" shows the percentage of users that rated the videos with 4 or 5. Bold values denote the most frequent user realism rating for each method and emotion.

I. Details about Running the Methods

For GANmut [4] and ICface [14], we use the codes made publicly available by the authors, whereas for DSM [13] we use the code that the authors provided us with. In all cases, we use the default parameters specified by the authors.

All methods were run on a machine with 4 NVIDIA RTX 2080 GPU. For training our *Neural Face Renderer*, we used Adam optimizer [8] with learning rate of 2e-4 for the meta-renderer or 4e-5 for the person-specific fine-tuning, β_1=0.5, β_2=0.999, and a batch size of 2. The *3D-based Emotion Manipulator* is trained with the Adam optimizer as well, using a learning rate of 1e-4, β_1=0, β_2=0.99 and a batch size of 64. The meta-renderer takes approximately 25 hours to complete training (15 epochs) using 4 GPUs, whereas fine-tuning an actor-specific renderer with 2 GPUs for 20

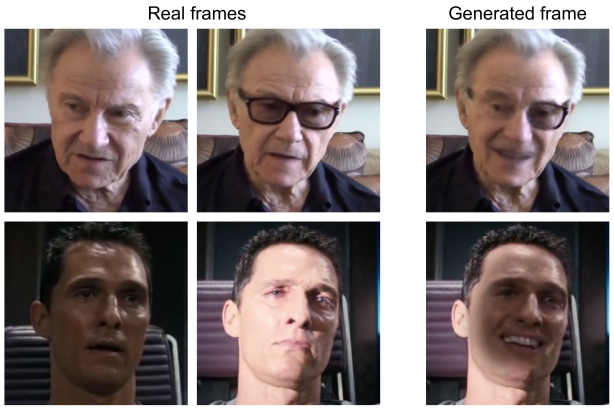


Figure 6. Typical failure cases of our *Neural Face Renderer* on challenging videos that include appearance and disappearance of glasses (1^{st} row) or diverse illumination conditions (2^{nd} row). The generator lacks conditional knowledge of which appearance/illumination to produce, thus generating a mixed result. The first 2 frames of each row correspond to real frames seen during training, while the rightmost frame corresponds to the middle frame being translated to *happy*.

epochs takes an average of 17 hours. The average end-to-end inference time corresponds to ~4 frames per second.

J. Neural Face Renderer failure cases

We note here that our face renderer assumes that the same foreground face is captured in a consistent scene throughout the whole duration of the training video. If this assumption does not hold (*e.g.* multiple faces appear in the foreground or the video contains diverse scenes with substantially different illumination conditions or actor's appearances in terms of makeup, facial hair, glasses etc), then our method will yield unrealistic results, like the mixed faces shown in Fig. 6.

References

[1] Peter J. Burt and Edward H. Adelson. A multiresolution spline with application to image mosaics. *ACM Trans. Graph.*, 2(4):217–236, Oct. 1983. 2

[2] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

[3] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3

[4] Stefano d'Apolito, Danda Pani Paudel, Zhiwu Huang, Andres Romero, and Luc Van Gool. Ganmut: Learning interpretable conditional space for gamut of emotions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 5, 6

[5] Michail Christos Doukas, Mohammad Rami Koujan, Viktoriia Sharmanska, Anastasios Roussos, and Stefanos Zafeiriou. Head2head++: Deep facial attributes re-targeting. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 3(1):31–43, 2021. 1

[6] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics (ToG), Proc. SIGGRAPH*, 40(4):88:1–88:13, 2021. 1

[7] Hyeongwoo Kim, Mohamed Elgharib, Michael Zollhöfer, Hans-Peter Seidel, Thabo Beeler, Christian Richardt, and Christian Theobalt. Neural style-preserving visual dubbing. *ACM Trans. Graph.*, 38(6), Nov. 2019. 3

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[9] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 1, 3

[10] X. Mao, Q. Li, H. Xie, R. K. Lau, Z. Wang, and S. Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2813–2821, 2017. 3

[11] Jacek Naruniec, Leonhard Helminger, Christopher Schroers, and Romann M. Weber. High-resolution neural face swapping for visual effects. *Computer Graphics Forum*, 39(4):173–184, 2020. 1

[12] Ivan Perov, Daiheng Gao, Nikolay Chervoniy, Kunlin Liu, Sugasa Marangonda, Chris Umé, Mr. Dpfks, Carl Shift Facenheim, Luis RP, Jian Jiang, Sheng Zhang, Pingyu Wu, Bo Zhou, and Weiming Zhang. Deepfacelab: Integrated, flexible and extensible face-swapping framework, 2021. 1

[13] Girish Kumar Solanki and Anastasios Roussos. Deep semantic manipulation of facial videos. *arXiv preprint arXiv:2111.07902*, 2021. 5, 6

[14] Soumya Tripathy, Juho Kannala, and Esa Rahtu. Icface: Interpretable and controllable face reenactment using gans. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020. 6

[15] Kaisiyuan Wang, Qianyi Wu, Linsen Song, Zhuoqian Yang, Wayne Wu, Chen Qian, Ran He, Yu Qiao, and Chen Change Loy. Mead: A large-scale audio-visual dataset for emotional talking-face generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 700–717, 2020. 3

[16] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017. 3