

# Supplementary Material

Alireza Javanmardi  
alireza.javanmardi@dfki.de  
Alain Pagani  
alain.pagani@dfki.de  
Didier Stricker  
didier.stricker@dfki.de

German Research Center for Artificial  
Intelligence (DFKI)  
Kaiserslautern, Germany

## 1 Architecture Design

In this section, we provide a detailed description of the G3FA model architecture for better understanding. We will focus on each module, starting with the keypoint extractor and concluding with the discriminator architecture. We will skip the theoretical aspects of all modules such as volume rendering, inverse rendering, and generative models, as they have been extensively covered in the referenced papers.

**Key point Detection:** To identify keypoints, we employ a U-net based [1] Autoencoder model as in [2], and shown in Fig. 2. This model takes an RGB image from the source or driving frame as input and produces heatmaps. By fitting a Gaussian function to the heatmap, we estimate the coordinates of the keypoints and their corresponding Jacobians. In our experiments, we select 15 keypoints and calculate  $2 \times 2$  Jacobians for each of them.

**Dense Motion Estimation:** We adopt the dense motion architecture proposed in [3]. As depicted in Fig. 3, we first obtain keypoints from both the source and driving images. To increase speed, we downsample the source image and perform sparse motion estimation based on the keypoints. Using the deformation map derived from this sparse motion, we deform the source image. We utilize a heatmap to capture the changes between the source and driving keypoints. Furthermore, we obtain masks for each keypoint, labeled from 0 to  $k$  as  $\{M_0, M_0, \dots, M_K\}$ , with the first mask specifically designed to ignore the background, while the remaining masks are utilized for warping source image features in subsequent steps as demonstrated in Fig. 1.

**3D Feature Extraction:** As demonstrated in Fig. 4, in order to render the animated face, we need to extract 3D features comprising shape and color information as proposed by [4]. We employ a 2D convolutional layer for color extraction and a ResBlock3D, which incorporates 3D convolutions with skip connections, to extract shape-related features in 3D.

**Orthogonal Adaptive Ray-Sampling:** In this module, depicted in Fig. 5, we leverage the 3D shape and color features to derive voxel probabilities and sample the color field. This

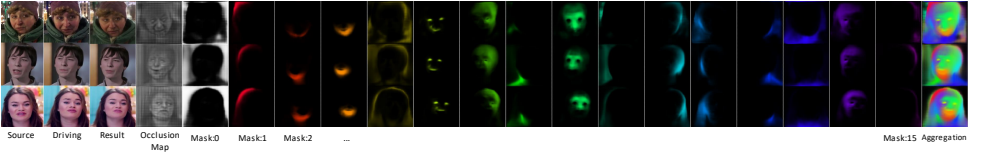


Figure 1: This figure visualizes the training process of our face animation model, achieved through self-supervised learning of occlusion map and weight masks.

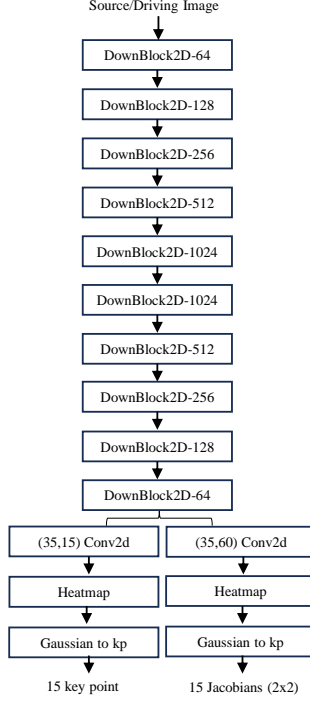


Figure 2: Architecture of keypoint extractor module.

module is implemented as a Multi-Layer Perceptron (MLP) as in [6] and plays a crucial role in the volume rendering process.

**Rendering Decoder:** Inspired by [6] and illustrated in Fig. 6, we concatenate warped source image features with the output of the volume rendering process, which results in a rendering feature map. This concatenated input is then passed through a series of SPADE layers[2] followed by 2D upsampling layers, enabling us to generate sharper rendering results. As shown in the figure, this decoding process significantly enhances the overall rendering quality.

**Discriminator:** For RGB, depth, and normal maps, we utilize the same discriminator architecture, except for the first layer where the depth map only has one channel. Following the approach described in [6], we incorporate a series of 2D Downblocks and Spectral Normalization[1] layers to generate the prediction map, as depicted in Fig. 7.

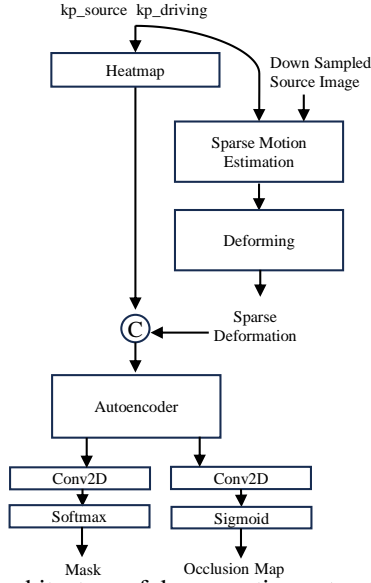


Figure 3: Architecture of dense motion extractor module.

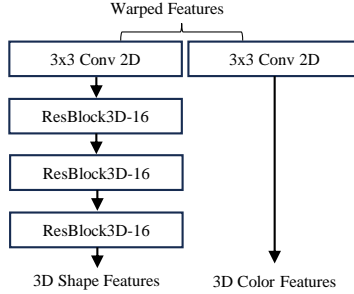


Figure 4: Architecture of 3D feature extractor module.

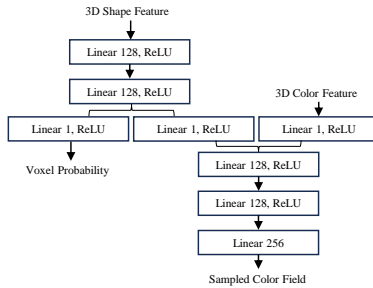


Figure 5: Architecture of orthogonal adaptive ray-sampler module.

In Table 1, we presented an analysis of the count of trainable parameters, along with the inference time measured in frames per second (FPS) on an RTX4090. As evident from

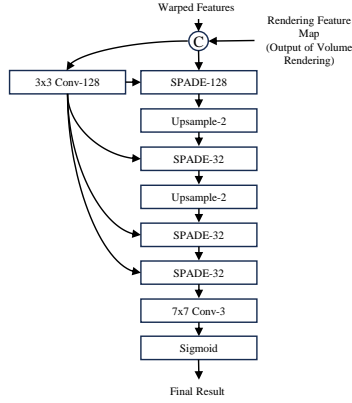


Figure 6: Architecture of rendering decoder module.

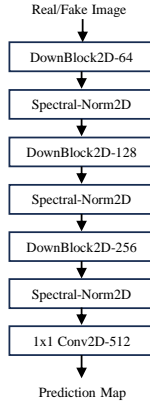


Figure 7: Architecture of discriminator module.

this table, G3FA exhibits a comparable number of parameters to FNeVR. However, G3FA delivers exceptional outcomes. Moreover, when compared with alternative methods, only FOMM and LIA achieve higher inference speeds, although they encounter issues with accurate reenactment. As shown in Table 2, the total number of trainable parameters in our model is comparable to that of FNeVR. The table highlights that each discriminator comprises only 4% of the total parameter count.

## 2 Ablation study

In order to validate the optimal configuration of our G3FA model, we conducted a comprehensive ablative analysis. Initially, we explored a scenario where a single discriminator was employed, with the depth image concatenated to the RGB input and presented as either real or generated data. However, this approach exhibited suboptimal convergence and manifested fluctuations across epochs. Consequently, as indicated in

Table 1: Quantitative assessment of model efficiency: Trainable parameters (M) and inference speed (FPS) analysis.

Method	Params (M)	FPS
FOMM	59.767	75.49
Face vid2vid	125.216	19.36
DaGAN	74.660	28.93
LIA	69.116	77.17
FNeVR	61.378	52.72
Ours	61.534	51.90

Table 2: Comparison of module parameters in G3FA

Model	Parameters(M)
Generator	39.041
RGB Discriminator	2.758
Depth Discriminator	2.756
Normal Discriminator	2.758
Keypoint Detector	14.522
Ours_total	61.564
FNeVR_total	61.378

Table 3: Ablation study of same-identity reconstruction on TK[8].

Method	L1 ↓	LPIPS ↓	PSNR ↑	SSIM ↑	FID ↓	AKD ↓
FNeVR baseline	8.12	0.097	24.47	0.69	19.56	1.74
G3FA + depth	8.04	0.092	25.77	<b>0.73</b>	18.47	1.66
G3FA + normal	8.10	0.095	24.32	0.70	20.93	1.70
G3FA (ours)	<b>7.89</b>	<b>0.084</b>	<b>27.26</b>	0.71	<b>16.41</b>	<b>1.62</b>

Table 3, we proceeded to evaluate our framework with one discriminator using RGB images and another discriminator employing either depth or normal data. The definitive version, which serves as the main architecture in our paper, incorporates both RGB and 3D information, employing separate discriminators, yielding superior outcomes. Notably, the determination of optimal discriminator weights, which we plan to combine, constituted an essential aspect of our framework. Through a series of experiments, we explored dynamic weighting schemes wherein  $\lambda_{depth}$  and  $\lambda_{normal}$  were incrementally increased every 10 epochs, or set to zero for initial epochs. However, none of these configurations yielded improved results.

To determine the optimal value of the hyperparameter  $\lambda$ , which determines the extent of incorporating 3D information in adversarial training, an experiment was conducted and the results are presented in Fig. 8 for three distinct values where  $\lambda = \lambda_{depth} = \lambda_{normal}$ . As evident from the graph, during the initial epochs, a higher value of  $\lambda$  accelerates the model’s convergence by demonstrating a significant reduction in the perceptual loss value. All discriminators, in this scenario, make comparable contributions to the adversarial loss due to their combination weight. When the rendering module falls short in generating photorealistic samples, the inverse rendering module likewise struggles to generate accurate

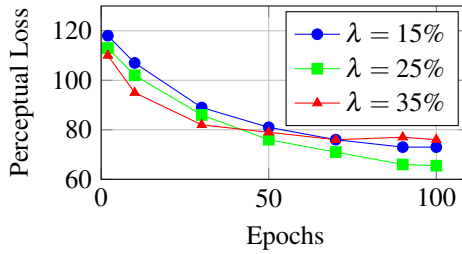


Figure 8: Perceptual loss values for various  $\lambda$  settings across 100 epochs on TK[5].

depth and normal maps. Consequently, these inaccuracies become distinguishable as fake samples for the discriminator. This compels the generator to significantly enhance its outputs to achieve the same quality as real samples. This will force the generator to produce much better results to be indistinguishable from real ones. Nevertheless, an excessive reliance on 3D information induces a decline in overall quality at the end of the training process. We selected  $\lambda = 25\%$ , as an intermediary value, striking a balance by leveraging the benefits of early model advancement while maintaining a sufficient emphasis on RGB values.

## References

- [1] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.
- [2] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization, 2019.
- [3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18*, pages 234–241. Springer, 2015.
- [4] Aliaksandr Siarohin, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, and Nicu Sebe. First order motion model for image animation. *Advances in Neural Information Processing Systems*, 32, 2019.
- [5] Ting-Chun Wang, Arun Mallya, and Ming-Yu Liu. One-shot free-view neural talking-head synthesis for video conferencing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10039–10049, 2021.
- [6] Bohan Zeng, Boyu Liu, Hong Li, Xuhui Liu, Jianzhuang Liu, Dapeng Chen, Wei Peng, and Baochang Zhang. Fnevr: Neural volume rendering for face animation. *Advances in Neural Information Processing Systems*, 35:22451–22462, 2022.