TriPlaneNet: An Encoder for EG3D Inversion

Ananta R. Bhattarai Matthias Nießner Artem Sevastopolsky

Technical University of Munich (TUM)

Abstract

Recent progress in NeRF-based GANs has introduced a number of approaches for high-resolution and high-fidelity generative modeling of human heads with a possibility for novel view rendering. At the same time, one must solve an inverse problem to be able to re-render or modify an existing image or video. Despite the success of universal optimization-based methods for 2D GAN inversion, those, applied to 3D GANs, may fail to produce 3D-consistent renderings. Fast encoder-based techniques, such as those developed for StyleGAN, may also be less appealing due to the lack of identity preservation. In our work, we introduce a real-time method that bridges the gap between the two approaches by directly utilizing the tri-plane representation introduced for EG3D generative model. In particular, we build upon a feed-forward convolutional encoder for the latent code and extend it with a fully-convolutional predictor of tri-plane numerical offsets. As shown in our work, the renderings are similar in quality to optimization-based techniques and significantly outperform the baselines for novel view. As we empirically prove, this is a consequence of directly operating in the tri-plane space, not in the GAN parameter space, while making use of an encoder-based trainable approach.

1. Introduction

Generative adversarial networks (GANs) have gained popularity due to their ability to generate images of high resolution and variation. In recent years, numerous works [11, 10] have tackled the problem of multi-view consistent image synthesis with 3D-aware GANs. Typically, nowadays these include a neural rendering engine and volumetrically integrate the internal 3D presentations. Such methods make generators aware of a 3D structure by modeling it with explicit voxel grids [20, 51, 38] or neural implicit representations [11, 41]. Most notably, EG3D [10] introduced a 3D GAN framework based on a tri-plane 3D representation that is both efficient and expressive to enable high-resolution 3D-aware image synthesis. Moreover, they demonstrate state-of-the-art results for unconditional geometry-aware image synthesis.

Figure 1: For a given picture, our method predicts the appropriate latent code and the tri-plane offsets for EG3D generator in a feed-forward manner. This way, both the frontal view and the novel view rendering can be obtained with high fidelity and in real-time.

Main applications of 3D GANs include human face inversion including head tracking, reenactment, facial manipulation, and novel view synthesis of a given image or video. Oftentimes, the classical GAN formulation does not support trivial inversion, i.e. finding the appropriate code in the learned GAN space for a given sample. A straightforward way to achieve this is by obtaining the latent code of the input image via optimization-based or encoder-based approaches, i.e. applying 2D GAN inversion techniques. An existing branch of research studies 2D GAN inversion in high detail [1, 45, 4, 2, 62, 52], but nevertheless, the problem remains underexplored in 3D.

Optimization-based inversion methods are often superior to encoder-based approaches in terms of reconstruction quality. However, encoder-based techniques are orders of

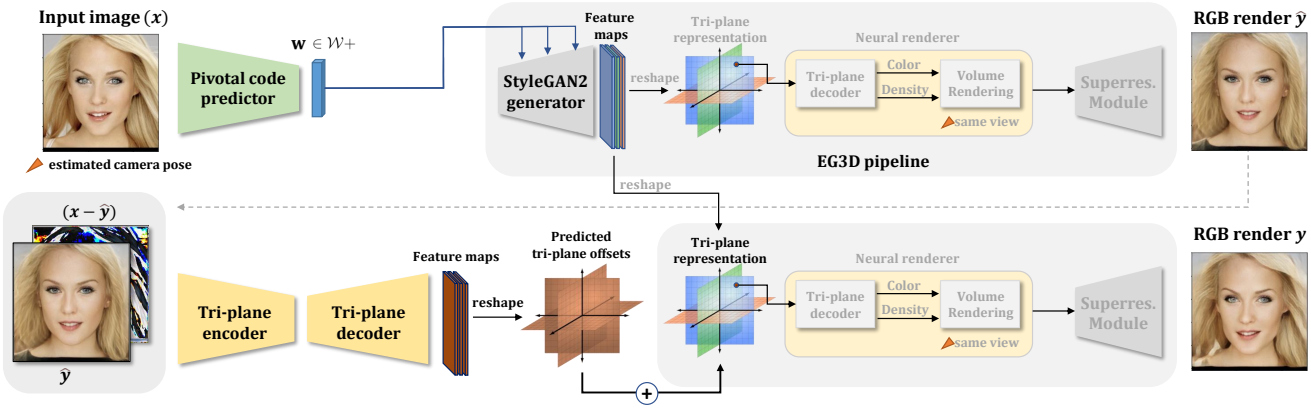


Figure 2: Overview of the proposed method. TriPlaneNet consists of two branches. The first branch (above) comprises the predictor $\hat{w} = \phi(x)$ of the pivotal latent code $\hat{w} \in \mathcal{W}+$, which results in an RGB image $\hat{y} = \mathcal{R}(\boldsymbol{G}(\hat{w}))$ after passing it through the EG3D tri-plane generator $\boldsymbol{G}(\cdot)$ and renderer block $\mathcal{R}(\cdot)$ containing super-resolution module. The second branch (below) uses the first-stage approximation $\hat{y}$ and its difference with the target $(x - \hat{y})$ to predict the numerical offsets to the tri-planes $\Delta\boldsymbol{T}$ by a convolutional autoencoder $\Delta\boldsymbol{T} = \psi(\hat{y}, x - \hat{y})$, which yields the final prediction $y = \mathcal{R}(\boldsymbol{G}(\hat{w}) + \Delta\boldsymbol{T})$.

magnitude faster as they map a given image to the latent space of GAN in a single forward pass. Compared to 2D GAN inversion, 3D GAN inversion is a more challenging task as the inversion needs to both preserve the identity of an input image and satisfy the 3D consistency constraint in generated novel views. In particular, optimization-based GAN inversion methods that have no knowledge of the specific GAN architecture make sure to yield a high-quality rendering of the desired image from the same camera view, but the lack of any geometry information in the image may produce broken or flattened geometry when rendered from a novel camera. We improve these shortcomings in two separate ways. First, by predicting an input latent code for the EG3D generator with a convolutional encoder, we observe that the geometry is preserved better. This can be attributed to the fact that the encoder, trained for the inversion task, is exposed to thousands of multi-view images and this way learns to be more 3D-aware. Second, we utilize the knowledge about the model and improve the fidelity and consistency by predicting offsets to the tri-planes that constitute the 3D representation in EG3D. Unlike voxel grids or implicit representations, tri-planes can be naturally estimated by 2D convnets and, as demonstrated by our experiments, can realistically express object features beyond the capabilities of an input latent code, e.g. hands and long hair, without breaking 3D consistency (see Fig. 1). This advantage is achieved by recovering the object representation directly in the world space. Since the tri-plane offsets are fully predicted by convolutional layers, our inversion can run in real-time on modern GPUs.

We propose the EG3D-specific inversion scheme in two stages. In the first stage, the initial inversion is obtained using the pSp encoder that directly embeds the input image into the $\mathcal{W}+$ space of EG3D. In the second stage, we introduce an encoder, TriPlaneNet, that learns to refine the reconstruction given the original image and initial inversion. Our second stage encoder takes the initial inversion, and the difference between initial inversion and the original image as the inputs and predicts a numerical offset to the initial tri-plane representation. Some recent works [5, 15, 46] for 2D inversion have attempted to improve the reconstruction quality in two stages. First, they obtain the latent code for the input image. Then, they fine-tune generator weights with respect to the initial inversion. However, we demonstrate that TriPlaneNet preserves identity and ensures 3D consistency in novel views better compared to other approaches.

To summarize, our contributions are the following:

- We propose a novel, real-time inversion framework for EG3D that enables high-quality reconstruction while maintaining multi-view consistency by directly utilizing the tri-plane representation.
- We demonstrate that our method achieves on par reconstruction quality compared to optimization-based inversion methods in the original view, while strongly outperforming them for novel view rendering. Our method is also more resilient towards harder cases such as when a hat or accessories are featured.

2. Related Work

3D Generative Models for Human Faces. Representing and generating diverse 3D human faces and heads attracted increasing attention over the last decade [37, 12, 23], while the appearance of NeRF [35] has sparked additional inter-

est in that topic. The first generative models built upon NeRF-style volumetric integration [48, 39] achieved generalization by conditioning the multi-layer perceptron on latent codes, representing the object's shape and appearance. More novel π-GAN [11] and StyleNeRF [17] condition the generative network on the output of the StyleGAN-like generator [26], which amounted to the higher-quality rendering of faces and arbitrary objects with subtle details. As a next major improvement step, authors of EG3D [10] propose a tri-plane 3D representation that serves as a bridge between expressive implicit representations and spatially-restricting explicit representations. As a byproduct, methods such as EG3D and StyleSDF [41] allow for the extraction of explicit, highly detailed geometry of the human faces, while being trained without any volumetric supervision. Some modifications of NeRFs [16, 42, 6] and NeRF-based generative models [8, 21] allows for the explicit expression and appearance control of the rendered faces. Further, recently demonstrated abilities of diffusion models to generate highly accurate 2D images are currently being transferred onto 3D objects [59, 36] and 3D human heads [54].

GAN Inversion. Unlike other kinds of generative models, such as VAE or normalizing flows, inverting a GAN (finding the appropriate latent code for a given image) is oftentimes a more tricky and computationally-demanding task. Early attempts focused on the tuning of the latent code with the optimization-based approaches [13, 30, 26]. Various approaches exploited the idea of predicting latent representation by an encoder [18, 34, 44, 63, 43]. In [46], a universal PTI method is introduced which comprises the optimization of a latent code and, consequently, fine-tuning parameters of the generator. A recent survey on GAN inversion [56] compares multiple generic techniques introduced since the appearance of GANs.

Inversion of 2D GANs For StyleGAN, an important observation was made by the authors of [1] that operating in the extended $\mathcal{W}+$ space is significantly more expressive than in the restrictive $\mathcal{W}$ generator input space. The latter idea has been strengthened and better adapted for face editing with the appearance of pSp [45] and e4e [53], as well as of their cascaded variant ReStyle [4] and other works [2, 62, 52]. Similarly to PTI but in an encoder-based setting, HyperStyle [5] and HyperInverter [15] predict offsets to the StyleGAN generator weights in a lightweight manner in order to represent the target picture in a broader space of parameters.

Inversion of 3D GANs. Unlike the 2D case, the inversion of a 3D GAN is a significantly more advanced problem due to the arising ambiguity: the latent code must be both compliant with the target image and correspond to its plausible 3D representation. While PTI remains a universal method that solves this problem for an arbitrary generator, recent art demonstrates the quality rapidly declines when the PTI inversion result is rendered from a novel view. The suggested ways of resolving this fidelity-consistency tradeoff for an arbitrary 3D GAN include incorporating multi-view consistency regularizers [29], augmenting training with surrogate mirrored images [57], or using depth information when available [28]. Our work employs a convolutional encoder and utilizes the EG3D tri-plane structure to improve the 3D consistency of the inverted rendering without additional multi-view constraints required during training.

Applications of GAN encoders. While being developed for inversion, convolutional encoders for GANs can compress a lot of information about the domain distribution and be used as a proxy. For instance, some of the recent works employ them for pre-training for semantic segmentation [7], face recognition [49], and as a generic prior [58, 40].

3. Method

3.1. Preliminaries

GAN inversion. Given a target image x, the goal of GAN inversion is to find a latent code that minimizes the reconstruction loss between the synthesized image and the target image:

$$\hat{w} = \underset{w}{\operatorname{argmin}}\, \mathcal{L}(x, G(w;\theta)) \tag{1}$$

where $G(w;\theta)$ is the image generated by pre-trained generator G parameterized by weights θ, over the latent w. The loss objective $\mathcal{L}$ usually employs L_2 or LPIPS [61]. The problem in (1) can be solved via optimization or encoder-based approaches. Encoder-based approaches utilize an encoder network E to map real images into a latent code. The training of an encoder network is performed over a large set of images $\{x^i\}_{i=1}^N$ to minimize:

$$\min_E \sum_{i=1}^{N} \mathcal{L}(x^i, G(E(x^i);\theta)) \tag{2}$$

During inference, an input image is inverted by $G(E(x);\theta)$. In the recent works [46, 5, 15], a number of approaches are proposed to additionally estimate image-specific generator parameters $\theta(x)$ by a convolutional network.

EG3D. EG3D [10] uses tri-plane 3D representation for geometry-aware image synthesis from 2D images. EG3D image generation pipeline consists of several modules: a StyleGAN2-based feature generator, a tri-plane representation, a lightweight neural decoder, a volume renderer, and a super-resolution module. To synthesize an image, a random latent code $z \in \mathbb{R}^D$ (typically, $D = 512$) and camera parameters are first mapped to a pivotal latent code $w \in \mathcal{W}+$ using a mapping network. Then, w is fed into the StyleGAN2 CNN generator $\boldsymbol{G}(\cdot)$ to generate a $H \times W \times 96$ feature map. This feature map is reshaped to form three 32-channel planes, thus forming a tri-plane feature representation $\boldsymbol{T}$ of the corresponding object. To

Figure 3: Qualitative comparison for image reconstruction. Compared to other approaches, our method can reconstruct a face in the same view in more detail, especially introducing more detail for features such as hats, hair, and background.

sample from the tri-plane features, a position $p \in \mathbb{R}^3$ is first projected onto the three feature planes. Then, corresponding feature vectors $(F_{xy}(p), F_{xz}(p), F_{yz}(p))$ are retrieved using bilinear interpolation and aggregated. These aggregated features are processed by a lightweight neural decoder to transform the feature into the estimated color and density at the location p. Volume rendering is then performed to project 3D feature volume into a feature image. Finally, a super-resolution module is utilized to upsample the feature image to the final image size. For simplicity, we will later refer to the lightweight neural decoder, renderer, and the super-resolution block, all combined, as the rendering block $\mathcal{R}(\cdot)$. The high efficiency and expressiveness of EG3D, as well as the ability to work with tri-planes directly, motivates the development of our model-specific inversion algorithm.

pSp. Richardson *et al*. [45] proposed a pSp framework based on an encoder that can directly map real images into $\mathcal{W}+$ latent space of StyleGAN. In pSp, an encoder backbone with a feature pyramid generates three levels of feature maps. The extracted feature maps are processed by a map2style network to extract styles. The styles are then fed into the generator network to synthesize an image $\hat{y}$:

$$\hat{y} = G(E(x) + \bar{w}), \tag{3}$$

where $G(\cdot)$ and $E(\cdot)$ denote the generator and encoder networks respectively and $\bar{w}$ is the average style vector of the pretrained generator.

3.2. TriPlaneNet

Our TriPlaneNet inversion framework comprises two branches (see Figure 2 for the overview). The first branch employs a pSp encoder to embed an input image into $\mathcal{W}+$ space of EG3D. Specifically, given an input image x, we train pSp encoder ϕ to predict the pivotal latent $\hat{w} \in \mathcal{W}+$:

$$\hat{w} = \phi(x) + \bar{w} \tag{4}$$

where the dimension of $\hat{w}$ is $K \times D$ (for the output image resolution of 128, $K = 14$, and $D = 512$). The piv-

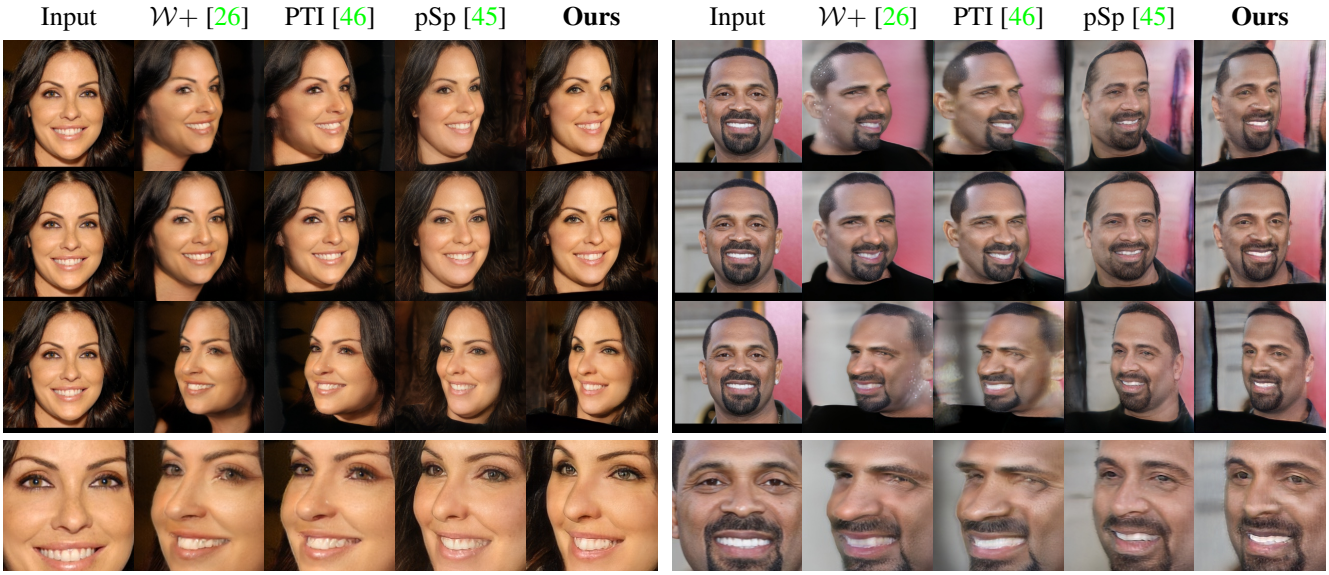


Figure 4: Qualitative evaluation on novel view rendering of yaw angle -0.6, -0.3, and 0.6 radians (full and zoom-in). In comparison to others, our method preserves identity and multi-view consistency better when rendered from a novel view.

otal code is then fed into StyleGAN2 generator $\boldsymbol{G}(\cdot)$ in the EG3D pipeline to obtain initial tri-plane features $\boldsymbol{T}$. Then, the tri-plane representation is processed by the rendering block $\mathcal{R}(\cdot)$ to generate initial reconstruction $\hat{y}$:

$$\hat{y} = \mathcal{R}(\boldsymbol{G}(\hat{w})) \tag{5}$$

The second branch consists of a convolutional auto-encoder ψ that learns to predict numerical offsets to the initial tri-plane features. The input to the autoencoder network is the channel-wise concatenation of initial reconstruction $\hat{y}$ and the difference between an input image and initial reconstruction $(x - \hat{y})$. Given this extended input, the autoencoder is tasked with computing tri-plane offsets $\Delta\boldsymbol{T}$ with respect to tri-plane features obtained in the first branch:

$$\Delta\boldsymbol{T} = \boldsymbol{\psi}(\hat{y}, x - \hat{y}) \tag{6}$$

The new tri-plane features corresponding to the inversion of the input image x are then computed as an element-wise addition of tri-plane offsets $\Delta\boldsymbol{T}$ with initial tri-plane features $\boldsymbol{T} = G(\hat{w})$. This new tri-plane representation is similarly processed by the rendering block $\mathcal{R}(\cdot)$ to obtain the final reconstructed image:

$$y = \mathcal{R}(\boldsymbol{T} + \Delta\boldsymbol{T}) \tag{7}$$

The autoencoder follows the typical U-Net [47] architecture, consisting of a contracting path and an expansive path. For our encoder backbone, we use the pre-trained IR-SE-50 architecture from [14]. In the decoder network, instead of using typical nearest neighbor upsampling, we use sub-pixel convolutional layers [50] in order to efficiently upscale the extracted features. The decoder architecture is similar to that of RUNet [22] with some minor modifications. Detailed overview of the architecture is presented in Appendix B.

3.3. Loss Functions

The pipeline is trained by minimizing the loss function that decomposes into the separate loss expressions for two branches:

$$\mathcal{L}_{\phi,\psi}(x, y, \hat{y}) = \mathcal{L}_{\phi}(x, \hat{y}) + \mathcal{L}_{\psi}(x, y) \tag{8}$$

For training the encoder $\phi(\cdot)$ in the first branch, we employ pixel-wise $\mathcal{L}_2$ loss, LPIPS loss [61], and ID loss [14], closely following the proposed loss for pSp [45]. Therefore, the total loss formulation is given by

$$\mathcal{L}_{\phi}(x, \hat{y}) = \lambda_1\mathcal{L}_2(x, \hat{y}) + \lambda_2\mathcal{L}_{\text{LPIPS}}(x, \hat{y}) + \lambda_3\mathcal{L}_{id}(x, \hat{y}) \tag{9}$$

In order to train the auto-encoder $\psi(\cdot)$ in the second branch, we replace $\mathcal{L}_2$ in (9) with $\mathcal{L}_1$ smooth loss. Then, the loss objective becomes:

$$\mathcal{L}_{\psi}(x, y) = \lambda_4\mathcal{L}_1(x, y) + \lambda_5\mathcal{L}_{\text{LPIPS}}(x, y) + \lambda_6\mathcal{L}_{id}(x, y) \tag{10}$$

4. Experiments

4.1. Training procedure

Datasets. Since our focus is on the human facial domain, we use FFHQ [25] dataset and 100,000 synthesized images from EG3D pre-trained on FFHQ for training and perform the evaluation on the 2824 CelebA-HQ [33, 24] test set. We extract the camera pose and pre-process the data in the same

Table 2: Quantitative comparison on image reconstruction. The inference time including the EG3D pass is given for a single RTX A100 Ti GPU. Our method is on par with PTI for frontal face inversion while being considerably faster.

Method	$L_2 \downarrow$	LPIPS $\downarrow$	ID $\uparrow$	MS-SSIM $\uparrow$	Infer. time $\downarrow$
$\mathcal{W}+$ [26]	0.08	0.18	0.72	0.78	77.07 s
PTI [46]	**0.011**	**0.07**	0.85	**0.90**	42.27 s
pSp [45]	0.049	0.18	0.69	0.70	**0.04 s**
Ours	0.016	0.09	**0.88**	0.89	0.07 s

way as in [10]. Since the pre-processing technique used couldn't identify the camera poses of 4 images, we skip the quantitative evaluation of 4 images for all the methods presented in the paper. Instead of using ground truth camera poses for synthesized images, we extract the camera pose following the same procedure as for other datasets to ensure consistency. We also augment the FFHQ dataset by mirroring it.

Training details. We conduct all our experiments in the human facial domain. Therefore, our pre-trained EG3D generator is also trained on the FFHQ dataset [25]. For training our models, we adopt the same training configuration from [45] except for some minor modifications, which will be explicitly mentioned here. We start the training of the second branch only after 20K steps and train both branches until 500K steps. Afterwards, we freeze the first branch and fine-tune the second branch until 1M steps. The model and the ablations are trained with a batch size of 3. We operate in the resolution of 256×256 and set the loss weights as follows: $\lambda_1 = \lambda_2 = 1$, $\lambda_3 = 0.1$, $\lambda_4 = 1$, $\lambda_5 = \lambda_6 = 0.1$.

Baselines. We compare our approach with both optimization- and encoder-based inversion methods. For optimization-based methods, we select $\mathcal{W}+$ optimization from [26] and PTI from [46]. For encoder-based methods, we choose pSp from [45]. For $\mathcal{W}+$ optimization, we optimize the latent code for 1K steps. For PTI, we first optimize the latent code $\hat{w} \in \mathcal{W}+$ for 1K steps and then fine-tune the generator for 1K steps. For pSp, we employ the original training configuration from [45] with a batch size of 4. In addition to FFHQ and mirrored FFHQ, we include 100K EG3D synthesized examples for training the pSp encoder.

4.2. Results

Comparison to the state-of-the-art. We present the evaluation of our approach w.r.t. the baselines in Fig. 3 and Table 2. Commonly used metrics L_2, LPIPS [61], MS-SSIM [55], and ID similarity [14] have been selected to analyze various aspects of perceptual similarity between inputs and corresponding reconstructions. Among others, PTI achieves the best results according to L_2, LPIPS, and MS-SSIM when the input image is re-rendered from the same view. However, our method outperforms the baselines according to the ID similarity, while being an order of magnitude times faster than optimization-based approaches. From a visual standpoint, our approach preserves finer details of the input image such as background objects, hair, and hat better than the other methods.

Table 3: Quantitative comparison on novel-view rendering. We significantly outperform the baselines by the ID score. L_2 and others not suitable here due to spatial misalignment.

Method	ID $\uparrow$						
	Novel View (Yaw angle in rad)						Avg.
	-0.8	-0.6	-0.3	0.3	0.6	0.8	
$\mathcal{W}+$ [26]	0.61	0.63	0.67	0.67	0.63	0.61	0.64
PTI [46]	0.65	0.69	0.76	0.76	0.69	0.66	0.70
pSp [45]	0.62	0.64	0.67	0.68	0.65	0.62	0.65
Ours	**0.70**	**0.74**	**0.83**	**0.83**	**0.76**	**0.72**	**0.76**

Furthermore, we demonstrate the identity preservation quality of input image re-rendering from a novel view in Table 3 and in Fig. 4. Our method significantly outperforms other inversion approaches w.r.t. ID similarity and extrapolates fine details to novel views in a more precise way. The ID score declines faster above a certain value of the yaw angle due to the limitation of EG3D to represent angles broader than in its training distribution.

Ablation study. In Fig. 5 and Table 4, we ablate over the possible differences in our model design, such as loss functions and changes introduced to the pSp [45] model. As some of those were introduced to mitigate checkerboard-style artifacts, we demonstrate visually how the LPIPS loss weight and incorporated pre-trained dual discriminator for EG3D affect the visual quality of the renderings. All models in this ablation are trained for 500K steps. We observed that setting LPIPS loss weight λ_5 to 1 introduces artifacts. As can be seen in Table 4, synthesized examples from EG3D for training improve both pSp and TriPlaneNet. As

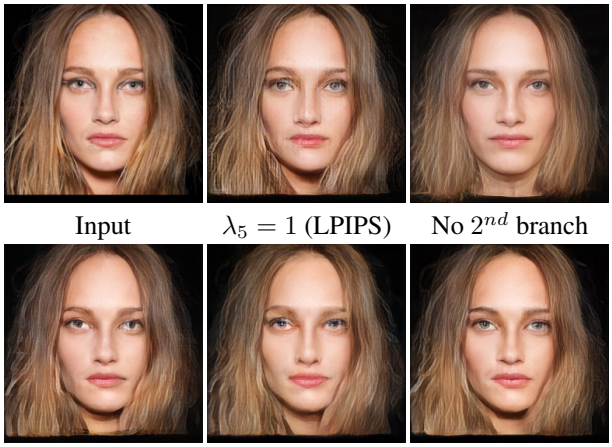
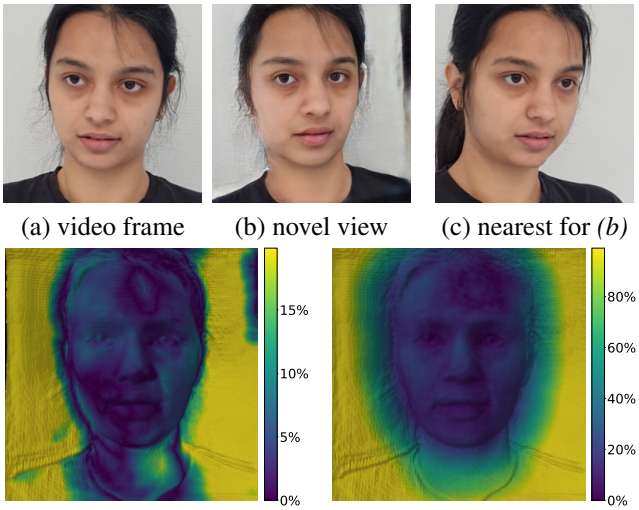


Figure 5: Qualitative ablation study for the loss, dataset, and architecture changes. $\mathcal{D}$ stands for the pre-trained EG3D dual discriminator [10] *Electronic zoom-in recommended.*

Table 4: Quantitative ablation study for the loss, dataset, and architecture changes.

Method	$L_2 \downarrow$	LPIPS $\downarrow$	MS-SSIM $\uparrow$	ID $\uparrow$						
				Same View	Novel View (Yaw angle in radians)					
					-0.8	-0.6	-0.3	0.3	0.6	0.8	Avg.
Ours	**0.018**	0.11	**0.87**	**0.84**	**0.68**	**0.72**	**0.79**	**0.80**	**0.73**	**0.69**	**0.73**
... no sub-pixel layers	0.019	0.11	0.87	**0.84**	**0.68**	**0.72**	**0.79**	**0.80**	**0.73**	**0.69**	**0.73**
... w/ discriminator	0.019	0.12	0.86	0.83	**0.68**	**0.72**	**0.79**	0.79	**0.73**	**0.69**	**0.73**
... $\lambda_5 = 1$ (LPIPS)	0.020	**0.09**	0.86	0.79	0.66	0.70	0.76	0.76	0.71	0.67	0.71
... w/o EG3D samples	0.021	0.13	0.86	0.83	0.67	0.71	0.78	0.78	0.72	0.67	0.72
pSp	0.076	0.22	0.59	0.70	0.63	0.65	0.68	0.69	0.66	0.63	0.66
... w/ EG3D samples	0.049	0.18	0.70	0.69	0.62	0.64	0.67	0.68	0.65	0.62	0.65



(a) video frame (b) novel view (c) nearest for *(b)*

(d) Recon. from view *(a)* vs. recon. from view *(c)*

(e) Recon. from view *(a)* vs. 3D model from SfM

Figure 6: For a sample frame *(a)* of a multi-view sequence, we encode it by TriPlaneNet and render from a novel view *(b)*, while *(c)* is the nearest to *(b)* frame in the sequence. In *(d)*, we compare the 3D reconstructions obtained by marching cubes from EG3D after encoding the frames *(a)* and *(c)* by nearest neighbor distance from *(a)* recon. to *(c)* recon. (in % of interocular distance). In *(e)*, the recon. from *(a)* view is compared to the 3D model estimated by SfM for the whole sequence, aligned to each other by face landmarks.

Figure 7: Novel view synthesis of frames extracted from a talking head video. *Electronic zoom-in recommended.*

demonstrated qualitatively and quantitatively, the best result is achieved by setting LPIPS loss weight to 0.1, including EG3D generated samples in the training set, using sub-pixel convolutional layers and disabling the discriminator.

Evaluation for a multi-view sequence. In Fig. 6, we evaluate the multi-view consistency of the reconstruction on a short smartphone-captured multi-view video of a person asked to stand still. The video consists of 106 frames and includes left-to-right sweeps approx. from $-90°$ to $90°$ yaw angle. We evaluate the novel view rendering capability for a single video frame by comparing it to the nearest neighbor video frame and the consistency of the marching cubes reconstructions from EG3D. Similarly, we measure the per-point distance to the 3D model estimated by SfM software [3] for the whole sequence. Despite that the shoulders and invisible head parts are hard to reconstruct for the method, we notice that the facial part and frontal hair are highly view-consistent and close to the ground truth.

Novel view synthesis for a talking head video. We demonstrate an application of our method to render in-the-wild videos from a novel view. In Fig. 7, frames of a video with a person talking and their rendering from a fixed novel view in the EG3D space are presented. The background in the video was removed by a matting network [27].

The encoder is capable of representing tiny details of in-the-wild portrait imagery in 3D and supports complex facial expressions.

4.3. PTI and tri-plane offsets behavior

Both our method and optimization- and encoder-based baselines can be decomposed into two stages: estimating the latent code and the delta for the generator parameters. In Fig. 8, we show how combining these steps, each performed either by optimization (opt.) or an encoder (pred.),

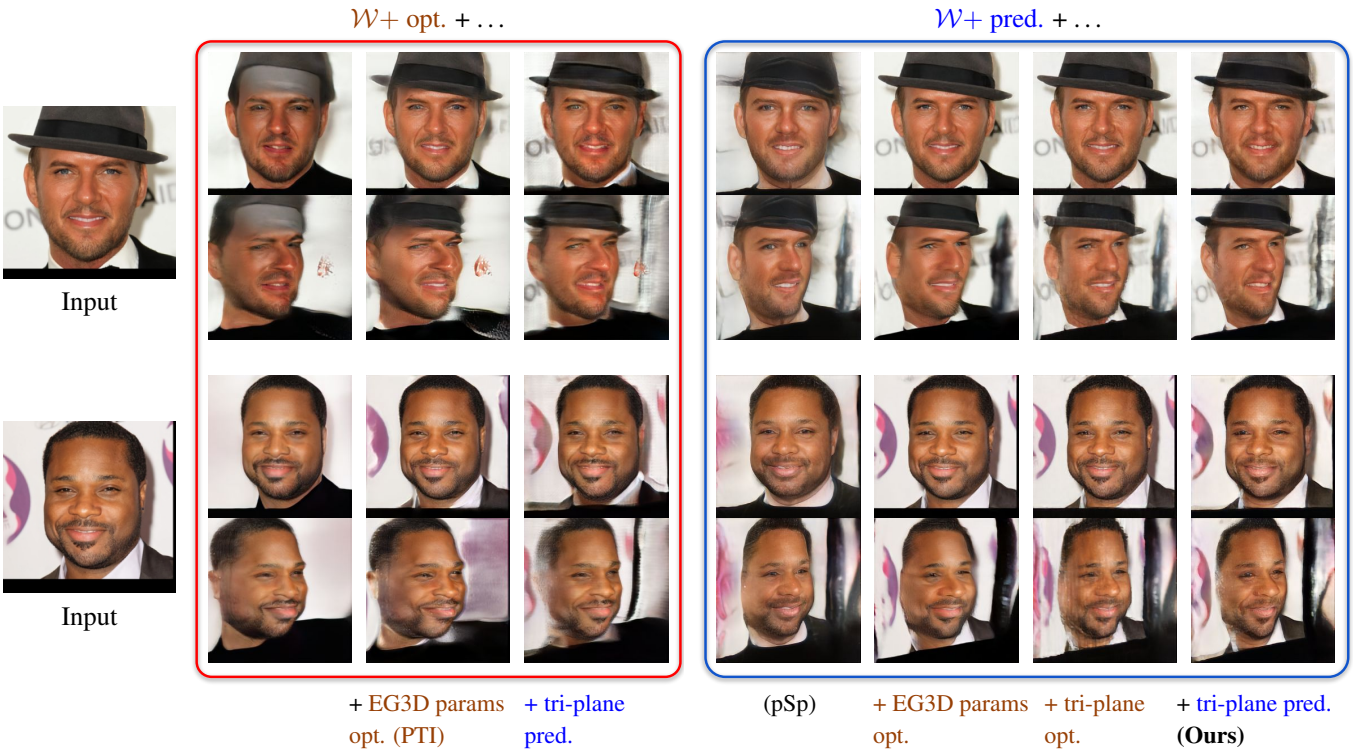


Figure 8: Comparison of hybrid approaches on CelebA-HQ test dataset. We refer to the computation done via optimization as opt. and via an encoder (pSp or TriPlaneNet) as pred. We observe that the methods starting from $\mathcal{W}+$ opt. have issues with consistency, whereas subsequent tri-plane pred. can partially alleviate it. Experiments starting from $\mathcal{W}+$ pred. demonstrate that the tri-plane space is more spatially restrictive than the EG3D parameters space. *Electronic zoom-in recommended.*

influences the inversion behavior.

$\mathcal{W}+$ opt. is performed per single image and, while it is capable of reconstructing the face in the same view, it doesn't account for 3D geometry due to the lack of the supervision from other views. As a result, we observe skewed rendering from a novel view. Accordingly, this is also the main reason why the consistency breaks with the PTI = ($\mathcal{W}+$ opt. + EG3D params opt.) method. Interestingly, tri-plane prediction, performed on top of $\mathcal{W}+$ opt., can alleviate the damage to the geometry caused by $\mathcal{W}+$ opt.

$\mathcal{W}+$ pred. by a pSp encoder, on the contrary, introduces more consistent geometry, since it was trained to reconstruct images from arbitrary views. At the same time, the rendering quality in the same view is marginally worse than PTI. Applying the PTI's second step (EG3D params opt.) helps improve it significantly, however, it incorrectly modifies head proportions, similar to the $\mathcal{W}+$ opt. behavior. To investigate this effect further, instead of optimizing EG3D parameters after $\mathcal{W}+$ pred., we try optimizing the tri-plane offsets directly, and this fully cancels the deterioration of geometry while preserving high fidelity in the same view. Since both EG3D params opt. and tri-plane opt. are performed for a single image (i.e. without multi-view supervision during training), this may indicate that offsetting the tri-planes is more spatially restrictive and thus stable. Our method reduces the artifacts observed with $\mathcal{W}+$ pred. + tri-plane opt. by utilizing an encoder for tri-plane offsets.

5. Conclusion

We present a novel approach for EG3D inversion that achieves high-quality reconstructions with view consistency and can be run in real time on modern GPUs. We also show that directly utilizing tri-plane representation better preserves 3D structure and identity in novel views compared to other approaches. Although our method achieves compelling results and is on par with optimization-based approaches for frontal face inversion, both visually and quantitatively, it has certain limitations. For instance, it is limited by the range of yaw angles shown to EG3D during training and cannot model the background depth. In addition, similarly to PTI and others, our approach may introduce artifacts for challenging examples in CelebA-HQ where the prediction of the first branch differs significantly from the input. We show a detailed analysis of such cases in Appendix C and propose a *cascaded test-time refinement* (CTTR) approach for improving the method's robustness.

Acknowledgments

This work was supported by the ERC Starting Grant Scan2CAD (804724) and the German Research Foundation (DFG) Research Unit "Learning and Simulation in Visual Computing." We also thank Yawar Siddiqui, Guy Gafni, Shivangi Aneja, and Simon Giebenhain for participating in the sample videos and helpful discussions.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4432–4441, 2019. 1, 3

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8296–8305, 2020. 1, 3

[3] Agisoft. *Metashape software. https://www.agisoft.com/.* 7

[4] Yuval Alaluf, Or Patashnik, and Daniel Cohen-Or. Restyle: A residual-based stylegan encoder via iterative refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6711–6720, 2021. 1, 3

[5] Yuval Alaluf, Omer Tov, Ron Mokady, Rinon Gal, and Amit Bermano. Hyperstyle: Stylegan inversion with hypernetworks for real image editing. In *Proceedings of the IEEE/CVF conference on computer Vision and pattern recognition*, pages 18511–18521, 2022. 2, 3

[6] ShahRukh Athar, Zexiang Xu, Kalyan Sunkavalli, Eli Shechtman, and Zhixin Shu. Rignerf: Fully controllable neural 3d portraits. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20364–20373, 2022. 3

[7] Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. *arXiv preprint arXiv:2112.03126*, 2021. 3

[8] Alexander W Bergman, Petr Kellnhofer, Yifan Wang, Eric R Chan, David B Lindell, and Gordon Wetzstein. Generative neural articulated radiance fields. *arXiv preprint arXiv:2206.14314*, 2022. 3

[9] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 12

[10] Eric R Chan, Connor Z Lin, Matthew A Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas J Guibas, Jonathan Tremblay, Sameh Khamis, et al. Efficient geometry-aware 3d generative adversarial networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16123–16133, 2022. 1, 3, 6, 24

[11] Eric R Chan, Marco Monteiro, Petr Kellnhofer, Jiajun Wu, and Gordon Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5799–5809, 2021. 1, 3

[12] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *Advances in neural information processing systems*, 29, 2016. 2

[13] Antonia Creswell and Anil Anthony Bharath. Inverting the generator of a generative adversarial network. *IEEE transactions on neural networks and learning systems*, 30(7):1967–1974, 2018. 3

[14] Jiankang Deng, J. Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4685–4694, 2018. 5, 6, 12

[15] Tan M Dinh, Anh Tuan Tran, Rang Nguyen, and Binh-Son Hua. Hyperinverter: Improving stylegan inversion via hypernetwork. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11389–11398, 2022. 2, 3

[16] Guy Gafni, Justus Thies, Michael Zollhofer, and Matthias Nießner. Dynamic neural radiance fields for monocular 4d facial avatar reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8649–8658, 2021. 3

[17] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. Stylenerf: A style-based 3d-aware generator for high-resolution image synthesis. *arXiv preprint arXiv:2110.08985*, 2021. 3

[18] Shanyan Guan, Ying Tai, Bingbing Ni, Feida Zhu, Feiyue Huang, and Xiaokang Yang. Collaborative learning for faster stylegan embedding. *arXiv preprint arXiv:2007.01758*, 2020. 3

[19] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015. 12

[20] Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. Escaping plato's cave: 3d shape from adversarial rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9984–9993, 2019. 1

[21] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20374–20384, 2022. 3

[22] Xiaodan Hu, Mohamed A. Naiel, Alexander Wong, Mark Lamm, and Paul Fieguth. Runet: A robust unet architecture for image super-resolution. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 505–507, 2019. 5, 12

[23] Rui Huang, Shu Zhang, Tianyu Li, and Ran He. Beyond face rotation: Global and local perception gan for photorealistic and identity preserving frontal view synthesis. In *Proceedings of the IEEE international conference on computer vision*, pages 2439–2448, 2017. 2

[24] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *ArXiv*, abs/1710.10196, 2017. 5

[25] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2018. 5, 6

[26] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 1, 3, 4, 5, 6, 14, 17, 18, 19, 20, 21, 22, 23, 25, 26

[27] Zhanghan Ke, Jiayu Sun, Kaican Li, Qiong Yan, and Rynson WH Lau. Modnet: Real-time trimap-free portrait matting via objective decomposition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1140–1147, 2022. 7

[28] Jaehoon Ko, Kyusun Cho, Daewon Choi, Kwangrok Ryoo, and Seungryong Kim. 3d gan inversion with pose optimization. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2967–2976, 2023. 3

[29] Yu-Jhe Li, Tao Xu, Bichen Wu, Ningyuan Zheng, Xiaoliang Dai, Albert Pumarola, Peizhao Zhang, Peter Vajda, and Kris Kitani. 3d-aware encoding for style-based neural radiance fields. *arXiv preprint arXiv:2211.06583*, 2022. 3

[30] Zachary C Lipton and Subarna Tripathi. Precise recovery of latent vectors from generative adversarial networks. *arXiv preprint arXiv:1702.04782*, 2017. 3

[31] Dong C Liu and Jorge Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1-3):503–528, 1989. 14

[32] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *ArXiv*, abs/1908.03265, 2019. 12

[33] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 3730–3738, 2014. 5

[34] Junyu Luo, Yong Xu, Chenwei Tang, and Jiancheng Lv. Learning inverse mapping by autoencoder based generative adversarial nets. In *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24*, pages 207–216. Springer, 2017. 3

[35] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[36] Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulò, Peter Kontschieder, and Matthias Nießner. Diffrf: Rendering-guided 3d radiance field diffusion. *arXiv preprint arXiv:2212.01206*, 2022. 3

[37] Thu Nguyen-Phuoc, Chuan Li, Lucas Theis, Christian Richardt, and Yong-Liang Yang. Hologan: Unsupervised learning of 3d representations from natural images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7588–7597, 2019. 2

[38] Thu H Nguyen-Phuoc, Christian Richardt, Long Mai, Yongliang Yang, and Niloy Mitra. Blockgan: Learning 3d object-aware scene representations from unlabelled images. *Advances in neural information processing systems*, 33:6767–6778, 2020. 1

[39] Michael Niemeyer and Andreas Geiger. Giraffe: Representing scenes as compositional generative neural feature fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 3

[40] Yotam Nitzan, Kfir Aberman, Qiurui He, Orly Liba, Michal Yarom, Yossi Gandelsman, Inbar Mosseri, Yael Pritch, and Daniel Cohen-Or. Mystyle: A personalized generative prior. *ACM Transactions on Graphics (TOG)*, 41(6):1–10, 2022. 3

[41] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. Stylesdf: High-resolution 3d-consistent image and geometry generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13503–13513, 2022. 1, 3

[42] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 3

[43] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019. 3

[44] Guim Perarnau, Joost Van De Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016. 3

[45] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2287–2296, 2021. 1, 3, 4, 5, 6, 12, 14, 15, 17, 18, 19, 20, 21, 22, 23, 25, 26

[46] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Transactions on Graphics (TOG)*, 42(1):1–13, 2022. 1, 2, 3, 4, 5, 6, 12, 14, 15, 17, 18, 19, 20, 21, 22, 23, 25, 26

[47] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *ArXiv*, abs/1505.04597, 2015. 5, 12

[48] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. *Advances in Neural Information Processing Systems*, 33:20154–20166, 2020. 3

[49] Artem Sevastopolsky, Yury Malkov, Nikita Durasov, Luisa Verdoliva, and Matthias Nießner. How to boost face recognition with stylegan? *arXiv preprint arXiv:2210.10090*, 2022. 3

[50] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan

Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1874–1883, 2016. 5

[51] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. Deepvoxels: Learning persistent 3d feature embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2437–2446, 2019. 1

[52] Ayush Tewari, Mohamed Elgharib, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt. Pie: Portrait image embedding for semantic control. *ACM Transactions on Graphics (TOG)*, 39(6):1–14, 2020. 1, 3

[53] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 40(4):1–14, 2021. 3

[54] Tengfei Wang, Bo Zhang, Ting Zhang, Shuyang Gu, Jianmin Bao, Tadas Baltrusaitis, Jingjing Shen, Dong Chen, Fang Wen, Qifeng Chen, et al. Rodin: A generative model for sculpting 3d digital avatars using diffusion. *arXiv preprint arXiv:2212.06135*, 2022. 3

[55] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, volume 2, pages 1398–1402 Vol.2, 2003. 6

[56] Weihao Xia, Yulun Zhang, Yujiu Yang, Jing-Hao Xue, Bolei Zhou, and Ming-Hsuan Yang. Gan inversion: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 3

[57] Fei Yin, Yong Zhang, Xuan Wang, Tengfei Wang, Xiaoyu Li, Yuan Gong, Yanbo Fan, Xiaodong Cun, Ying Shan, Cengiz Oztireli, et al. 3d gan inversion with facial symmetry prior. *arXiv preprint arXiv:2211.16927*, 2022. 3

[58] Jiawei You, Ganyu Huang, Tianyuan Han, Haoze Yang, and Liping Shen. A unified framework from face image restoration to data augmentation using generative prior. *IEEE Access*, 11:2907–2919, 2023. 3

[59] Xiaohui Zeng, Arash Vahdat, Francis Williams, Zan Gojcic, Or Litany, Sanja Fidler, and Karsten Kreis. Lion: Latent point diffusion models for 3d shape generation. *arXiv preprint arXiv:2210.06978*, 2022. 3

[60] Michael Ruogu Zhang, James Lucas, Geoffrey E. Hinton, and Jimmy Ba. Lookahead optimizer: k steps forward, 1 step back. *ArXiv*, abs/1907.08610, 2019. 12

[61] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018. 3, 5, 6, 14

[62] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, pages 592–608. Springer, 2020. 1, 3

[63] Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A Efros. Generative visual manipulation on the natural image manifold. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part V 14*, pages 597–613. Springer, 2016. 3

A. Training details

We operate in the resolution of 256×256 except for the calculation of $\mathcal{L}_{id}$. Specifically, the region around the face is cropped and resized to 112×112 before feeding into the face recognition network [14] to calculate $\mathcal{L}_{id}$. To train our models, we use the Ranger optimizer that combines Rectified Adam [32] with the Lookahead technique [60] and set a learning rate to 0.0001. The model is trained using a single NVIDIA GeForce RTX 3090 GPU.

A.1. EG3D samples added to the training data set

Our model is trained on a combination of real images from FFHQ and generated samples from EG3D. As shown in the ablation study in the main text, adding EG3D samples helps both pSp [46] and TriPlaneNet. The synthetic training samples are generated via sampling latent codes z for EG3D with no truncation ($\psi = 1$) often applied for large-scale GANs [9], thus including the hard samples. Figure 10 shows the comparison between FFHQ and EG3D samples train losses for $\psi = 1$ and $\psi = 0.7$. We attribute the difference between the curves with $\psi = 0.7$ to the domain gap between the generated samples from EG3D and real ones from FFHQ. In order to match the camera pose distribution in the FFHQ, we randomly sample FFHQ camera poses including mirrored while generating EG3D samples.

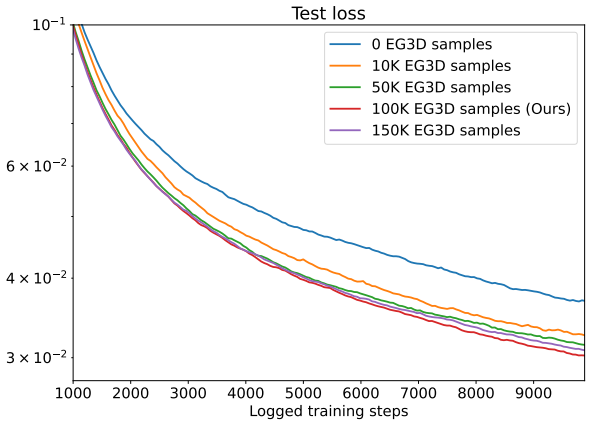


Figure 9: Test loss (calculated over the CelebA-HQ) during training for experiments with varying number of EG3D samples. The choice of 100K EG3D samples turns out to be optimal w.r.t. the test loss closer to the end of training.

In Table 5, we demonstrate the dependence of the reconstruction quality on CelebA-HQ on the number of synthetic samples, created by EG3D in advance added to the dataset. Although the change in metrics is marginal with respect to the increasing number of EG3D samples, test loss shown in Figure 9 demonstrates that the amount of 100K samples, used for the final model, is close to optimal.

B. Architecture details

First Branch To implement the encoder, we adopt the design of the pSp encoder from [45]. As the EG3D generator expects 14 styles vector for the selected resolution, we modify the pSp architecture to output 14 styles vector instead of 18. For the backbone network, we employ IR-SE-50 [14] pretrained for face recognition.

Second Branch (TriPlaneNet) Our TriPlaneNet consists of an encoder and a decoder network, a typical U-Net [47] architecture. The encoder backbone is an IR-SE-50 [14] pretrained on face recognition, which accelerates convergence. We adopt the design of the RUNet [22] for the decoder with some minor modifications. Instead of using ReLU as in RUNet, we use PReLU [19] with a separate α for each input channel and an initial value of 0.25. Similar to RUNet, every step in the decoder path consists of upsampling, concatenation, and convolution operations. Upsamping of the feature map is performed with a sub-pixel convolutional layer. Then, it is followed by a concatenation with the intermediate feature maps from the encoder path. The intermediate features are extracted from the 2nd, 6th, 20th, and 21st layers of the encoder. Finally, batch normalization, 3×3 convolution, PReLU, 3×3 convolution, and PReLU are applied sequentially. The last step in the decoder path comprises sub-pixel convolutional, 3×3 convolution, PReLU, 3×3 convolution, PReLU, and 1×1 convolution operations, while result in $256 \times 256 \times 96$ tri-plane offsets in the end.

C. Cascaded Test-Time Refinement (CTTR)

We propose a Cascaded Test-Time Refinement (CTTR) procedure to refine the reconstructions for harder cases (see Figure 11 for the overview). At the inference time, given a hard input image x and final image y reconstructed by the second branch of TriPlaneNet, we reuse the second branch by passing y as a replacement for the previous $\hat{y}$ in the second branch input. This provides us the refined tri-plane offsets $\Delta \boldsymbol{T_r}$:

$$\Delta \boldsymbol{T_r} = \boldsymbol{\psi}(y, x - y) \tag{11}$$

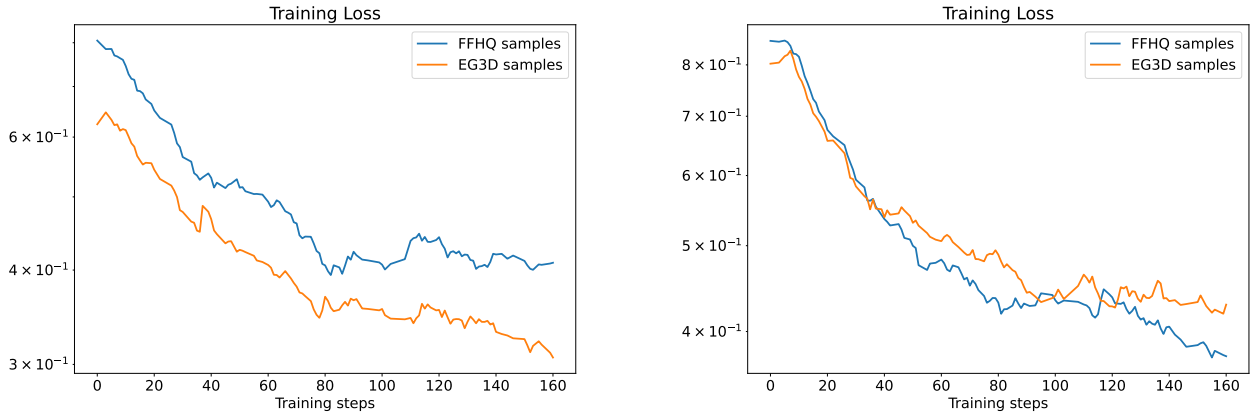
This $\Delta \boldsymbol{T_r}$ is summed with the initial tri-plane features $\boldsymbol{T} = G(\hat{w})$ and passed through the rendering block $\mathcal{R}(\cdot)$ to obtain the new refined image:

$$y_r = \mathcal{R}(\boldsymbol{T} + \Delta \boldsymbol{T_r}) \tag{12}$$

The same procedure can be applied N times in a cascaded way by reusing the image y_r as a new input to the second branch. However, we found that applying it only once is a reasonable trade-off between quality and the proportionally increased inference time. We show how each additional step of CCTR affects the reconstruction quality and increases

Table 5: Quantitative ablation study over the number of synthesized EG3D samples in the training set. We do not highlight any specific numbers in bold to demonstrate that the difference between {10K, ..., 100K} is relatively negligible. Due to that, the number of EG3D samples was selected according to the least observed overfitting (see Fig. 9).

Our Method	$L_2 \downarrow$	LPIPS $\downarrow$	MS-SSIM $\uparrow$	ID $\uparrow$						
				Same View	Novel View (Yaw angle in radians)					
					-0.8	-0.6	-0.3	0.3	0.6	0.8	Avg.
... w/ 0 EG3D samples	0.021	0.13	0.86	0.83	0.67	0.71	0.78	0.78	0.72	0.67	0.72
... w/ 10K EG3D samples	0.019	0.11	0.87	0.85	0.68	0.72	0.80	0.80	0.74	0.69	0.74
... w/ 25K EG3D samples	0.018	0.11	0.87	0.84	0.68	0.72	0.79	0.80	0.74	0.69	0.74
... w/ 50K EG3D samples	0.019	0.12	0.87	0.85	0.69	0.73	0.80	0.81	0.74	0.70	0.74
... w/ 75K EG3D samples	0.018	0.12	0.87	0.83	0.67	0.71	0.78	0.79	0.73	0.69	0.73
... w/ 100K EG3D samples	0.018	0.11	0.87	0.84	0.68	0.72	0.79	0.80	0.73	0.69	0.73



(a) $\psi = 0.7$ (b) $\psi = 1.0$ (Ours)

Figure 10: Training curves for two values of the truncation multiplier ψ defining the difficulty of the synthetic samples. The loss for the batches consisting fully of FFHQ and fully of EG3D samples is reported separately to demonstrate the consequences of selecting ψ on the observed domain gap between the real and synthetic data.

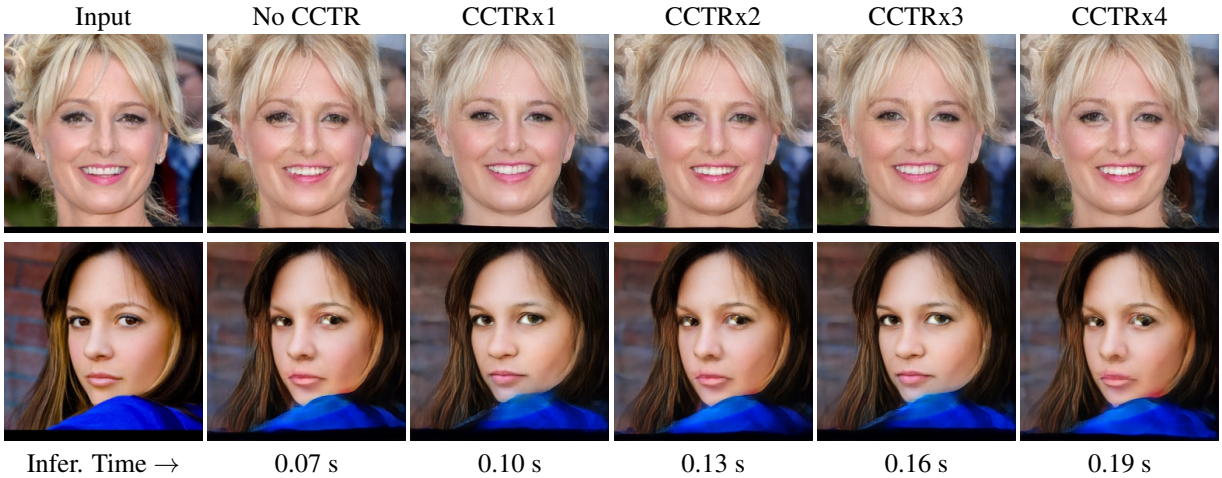


Figure 12: Comparison of reconstruction quality and inference time without and with each additional step of CCTR. Since single CCTR step brings the image closer to the pSp (first branch) output in terms of the plausibility, CTTRx{1,3} and CTTRx{0,2,4} look similar.

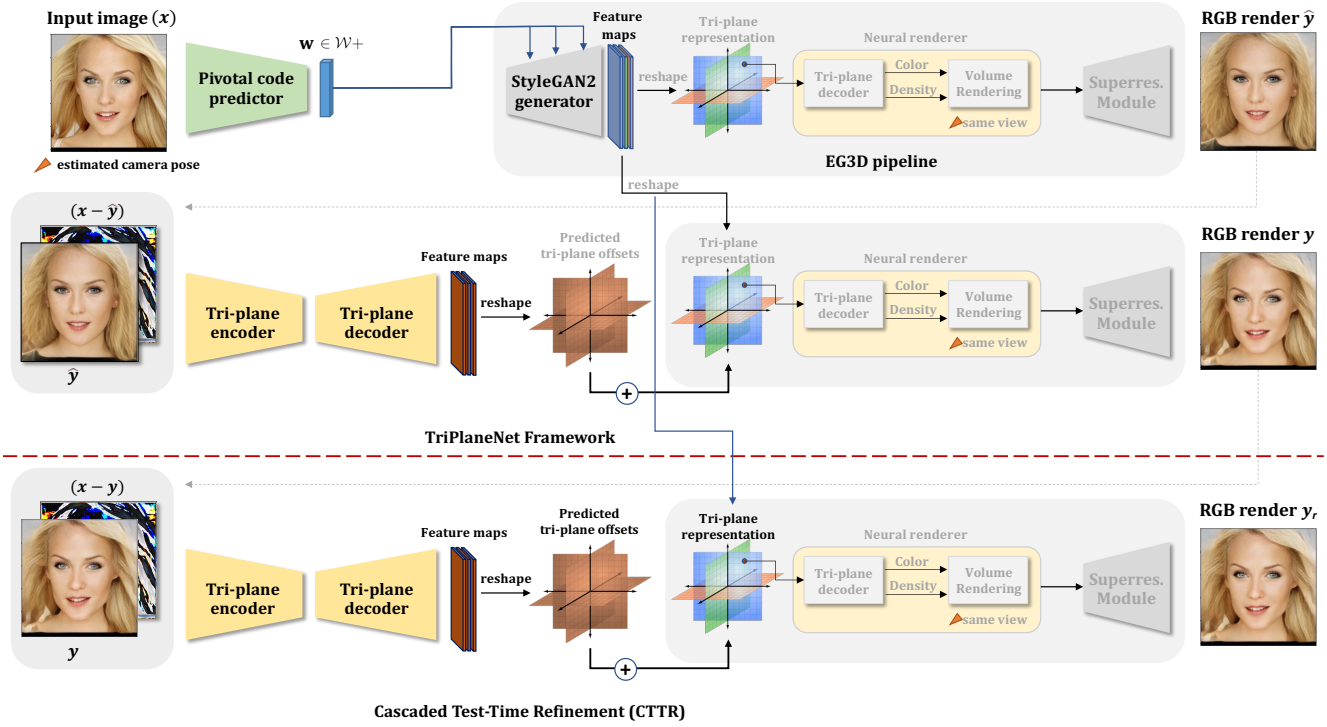


Figure 11: Overview of the Cascaded Test-Time Refinement (CTTR). In the CCTR step, the input image x and reconstructed image y is passed once again through the already trained second branch of TriPlaneNet to obtain the refined tri-plane offsets $\Delta \boldsymbol{T}_r$. The refined tri-offsets are added to the initial tri-plane features $\boldsymbol{T}$, which are then processed by the renderer block $\mathcal{R}(\cdot)$ to yield the refined reconstruction $y_r = \mathcal{R}(\boldsymbol{G}(\hat{w}) + \Delta \boldsymbol{T}_r)$.

.

the inference time in Figure 12. Figure 23 and 24 shows the visual comparison before and after the refinement for two groups of examples: *easy samples,* the reconstruction before refinement is plausible and close to the input, and *hard samples,* where some regions, e.g. eyes, feature significant artifacts prior to the refinement. For easy samples, we observe that refinement removes very fine details and does not help with such cases. However, for hard samples, even though refinement does not fully preserve the input image, it creates more plausible images and removes artifacts. Therefore, we recommend applying the refinement approach in more challenging cases.

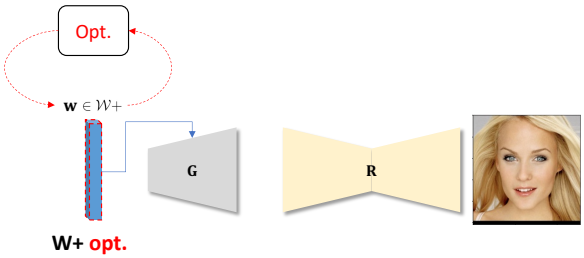
D. Novel view rendering of videos

We demonstrate additional results in Figure 14 for novel view rendering of in-the-wild videos.

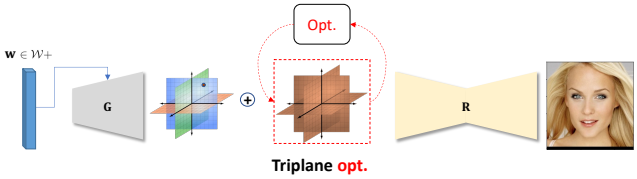
E. Discussion of the baselines design

In Figure 13, we provide visual overview of the baseline designs used for the analysis of PTI [46] and tri-plane offsets behavior. One-stage inversion techniques can be divided into two approaches: optimization-based and based on an encoder prediction. Two-stage inversion techniques involve inference followed by fine-tuning of some of the generator parameters. Similarly, these parameters can be fine-tuned via optimization or by encoder prediction. Since our method involves prediction of the tri-plane offsets and avoid fine-tuning of the generator parameters, we also consider the baseline where the tri-plane offsets in the second stage are optimized. To furthermore understand PTI [46] and the tri-plane offsets behavior, we design different hybrid two-stage inversion approaches that combine both optimization and prediction.

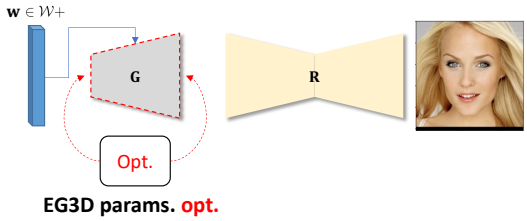
For $\mathcal{W}+$ opt., we optimize the latent code $w \in \mathcal{W}+$ for 1K steps following [26]. The $\mathcal{W}+$ pred. constitutes the baseline with the latent code $w \in \mathcal{W}+$ predicted by pSp encoder [45]. For EG3D params opt., we apply the second stage of PTI from [46] and optimize for 1K steps. To optimize for the tri-plane offsets (tri-plane opt.), we use L-BFGS [31] as the optimizer and employ combination of L_2 or LPIPS [61] with regularization term (L_2 and LPIPS discrepancy with the first branch prediction) as a loss objective. We run the optimization for 50 steps. As L-BFGS approximates the Hessian by calculating several estimates in a single step, 50 steps is equivalent to optimizing EG3D



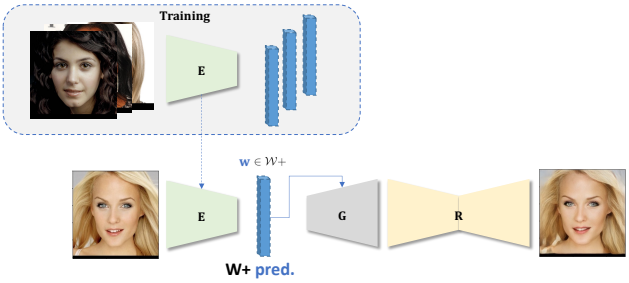
(a) $\mathcal{W}+$ opt.



(b) Tri-plane opt.



(c) EG3D params opt.



(d) $\mathcal{W}+$ pred. (pSp [45])

Figure 13: Overview of our baseline approaches. $\boldsymbol{G}(\cdot)$ and $\mathcal{R}(\cdot)$ stand for EG3D generator and renderer blocks respectively. Hybrid approaches described in the main text constitute the combination of the techniques shown above, applied sequentially one after another. For instance, PTI [46] sequentially performs (a) and then (c)., and $\mathcal{W}+$ pred. + tri-plane opt. sequentially performs (d) and then (b).

params for 1K steps.

F. Additional Qualitative Results

In this section, we present additional qualitative results on same view inversion, novel view rendering and ablation for the loss, dataset, and architecture changes.

- Figures 15 and 16 provide qualitative comparison of our approach with existing state-of-the-art inversion techniques for image reconstruction.
- Figures 17, 18, 19, 20, and 21 demonstrate qualitative comparison of our approach with existing state-of-the-art inversion techniques on novel-view rendering.
- Figure 22 contains extensive qualitative ablation study for the loss, dataset, and architecture changes.

Figure 14: More novel view synthesis of frames extracted from a talking head video. The novel view yaw angles from the frontal are as follows: second row (0.3 radians), fourth row (0.6 radians), sixth row (0.3 radians), and last row (0.8 radians). *Electronic zoom-in recommended.*

Figure 15: Additional qualitative comparison for image reconstruction. The last row additionally demonstrates the results with the proposed refinement procedure (CTTR) which is only required for the challenging cases (see Fig. 24).

Figure 16: Additional qualitative comparison for image reconstruction. The last row additionally demonstrates the results with the proposed refinement procedure (CTTR) which is only required for the challenging cases (see Fig. 24).

Figure 17: Additional qualitative evaluation on novel view rendering of yaw angle -0.6, -0.3, 0.3, and 0.6 radians.

Figure 18: Additional qualitative evaluation on novel view rendering of yaw angle -0.6, -0.3, 0.3, and 0.6 radians.

Figure 19: Additional qualitative evaluation on novel view rendering of yaw angle -0.6, -0.3, 0.3, and 0.6 radians.

Figure 20: Additional qualitative evaluation on novel view rendering of yaw angle -0.6, -0.3, 0.3, and 0.6 radians.

Figure 21: Additional qualitative evaluation on novel view rendering of yaw angle -0.6, -0.3, 0.3, and 0.6 radians.

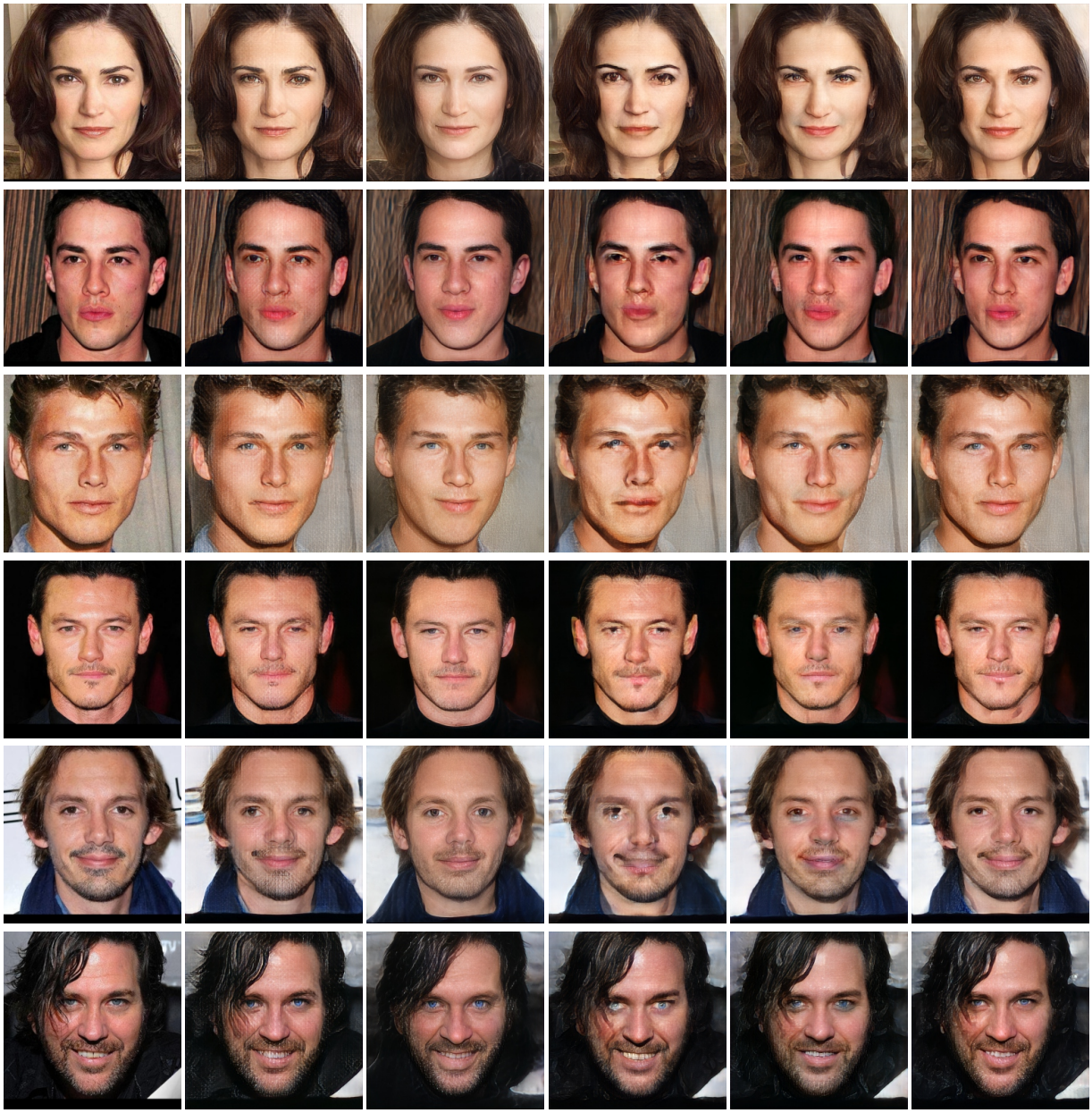


Figure 22: Additional qualitative ablation study for the loss, dataset, and architecture changes. $\mathcal{D}$ stands for the pre-trained EG3D dual discriminator [10].

Figure 23: Qualitative comparison for image reconstruction on easy cases. Our proposed CTTR approach (last row) does not have much effect for well-reconstructed examples with our inversion framework (TriPlaneNet).

Figure 24: Qualitative comparison for image reconstruction on challenging cases. We observe that some of the face regions, e.g. eyes, look more plausible with the CTTR procedure (last row) in these cases, at the cost of slightly less preserved identity and increased inference time.