

COSY: Compositional 3DGS Synthesis for Disentangled Human Head Editing

Florian Barthel^{1,2}, Shalini De Mello³, Koki Nagano³, Wieland Morgenstern², Anna Hilsmann¹, and Peter Eisert¹

¹ Fraunhofer Heinrich-Hertz Institute, Germany

² Humboldt University Berlin, Germany

³ NVIDIA, USA

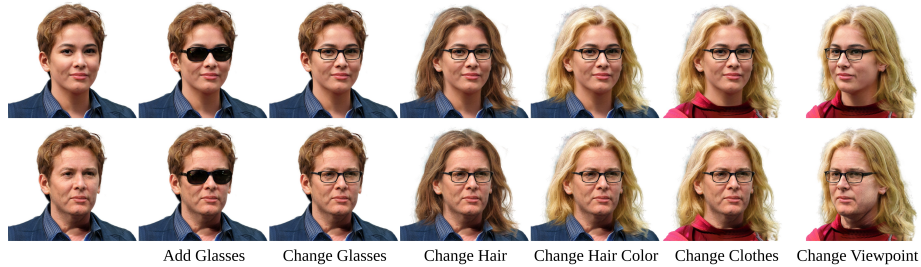


Fig. 1: Our method provides an independent latent space for hair, face, glasses, and torso, allowing precise and disentangled editing of 3D human head avatars (Project).

Abstract. Recent 3D Gaussian Splatting (3DGS) GANs for human heads synthesize and render photorealistic 3D models in real-time and offer a vast variety in identity and appearance. However, controlling specific semantic attributes such as hair color or glasses remains challenging, as edits in the entangled latent space often induce unintended changes in identity or appearance. Although there are several methods that aim to disentangle the latent space post training by estimating directions that only modify certain features, these methods cannot guarantee complete disentanglement and often require pre-trained classifiers. In our approach, we propose a new generator architecture that synthesizes components, such as hair, skin, glasses, and torso, completely independently. This allows for changing the latent vector for one region while keeping the remaining parts fixed. Further, we achieve this separation using only sparse information such as the hair or skin color, eliminating the requirement of segmentation masks or geometric priors, often seen in prior work. To ensure matching shape and lighting conditions during editing, we allow minimal shared information via context tokens between the independent generators. These tokens even allow us to control the shape and light, without any prior annotation. Compared to existing works on GAN-based generation and editing, our method shows better disentanglement, more precise editing control, and competitive visual quality.

1 Introduction

The synthesis of realistic 3D human heads is fundamental to film, games, virtual reality, robotics, and interactive content creation. These applications require controllable and disentangled model generation, enabling the user to explore and edit the appearance such as hairstyle or accessories in an interactive manner while preserving the subject’s identity and 3D structure. Controlled 3D synthesis also benefits digital twin pipelines and synthetic data generation (SDG), which depend on scalable methods for producing diverse image, 3D assets and label pairs to support downstream learning and simulation tasks.

Creating and editing high-quality 3D avatar heads, however, often requires extensive manual effort or expensive capturing studios. For this reason, there is a high demand for automating the process while keeping costs to a minimum. 3D GANs have emerged as a suitable framework for this task, as they learn explicit 3D representations exclusively from 2D training data. This makes them stand out from recent diffusion models, which primarily target 2D and video generation or require 3D ground truth data as supervision. Additionally, 3D GANs achieve real time generation capabilities and provide a smooth latent space, that facilitates editing the appearance, also via GAN inversion, which estimates a latent vector that re-synthesizes a specific identity from a given target image. This allows to reconstruct and edit the 3D representation of an identity from a single 2D image.

Although the rendering quality of 3D GANs has improved substantially with recent advances in NeRF [2, 7, 38, 40] and 3DGS [3, 4, 17, 22, 26, 43] based GANs, it still remains challenging to exclusively edit specific attributes of the generated subjects without unintentionally changing other aspects. The core difficulty lies in the unconditional latent space of the generator, in which all semantics are strongly entangled. Although the generator implicitly organizes the latent space during training, enabling editing control through the identification of latent directions corresponding to image attributes [18], we still observe strong entanglement between the attributes. Another reason for this strong entanglement are biases in the limited training data. For example, in real-world datasets, hair and skin color are highly correlated, making it very difficult to separate these features during editing.

To address these limitations, we propose a new compositional generator architecture that explicitly separates 3D head synthesis into semantically localized components. Our approach uses multiple sub-generators, each responsible for the generation of a specific component, i.e. hair, skin, glasses, and torso. This enables independent control of semantic attributes by modifying the latent vector for a single sub-generator while keeping all others fixed. The separation of components is completely implicit without relying on explicit segmentation masks or geometric priors. Further, as we randomly mix components already during the training, we enforce the generator to produce compatible components. In order to encode global attributes such as overall head size and lighting direction, we provide all generators with shared learned features, which even allow us to control the shape and lighting without using any prior annotation.

Even though training multiple coordinated sub-generators is more challenging compared to generators with a single output, our approach achieves competitive visual quality, while ensuring structurally disentangled editing of head components. Compared to prior work on real-time GAN editing, our work is the first to disentangle the 3D head components using implicit annotations, thus allowing for precise and real-time editing at high resolution. The contributions of our work are:

1. A 3DGS GAN architecture that decomposes the synthesis into semantically localized sub-generators, without explicit annotation.
2. Shared shape and lighting context latent vectors ensuring consistent head geometry and global illumination.
3. Training strategies to improve the quality of novel component combinations.

2 Related Work

Generative Adversarial Networks (GANs) [13] perform a non-cooperative game between a generator that synthesizes images from random latent vectors and a discriminator that aims to differentiate between generated images and real images from a dataset. During training, both networks are optimized until the generated images look indistinguishable from real images. To better control the synthesized output, conditional GANs [31] allow to include annotations to the training process. Typically, the generator uses the input label to condition the latent vector in order to produce the respective output. To provide conditional feedback, the same label is forwarded to the discriminator, which then decides whether the image is real or fake given the input label.

3D-aware GANs 3D-aware GANs are a variant of conditional GANs that use camera parameters as conditioning. However, instead of encoding the latent vector with the camera during generation, the camera parameters are applied to render an intermediate 3D representation produced by the generator. This allows 3D-aware GANs to learn 3D scenes from in-the-wild single-view images. One of the first 3D-aware GANs was π -GAN [8], which generates the weights of a NeRF [30] MLP. While producing good results, rendering with NeRFs is very slow. As a result, EG3D [7] and GRAM [40] proposed more efficient 3D representations that store 3D features in a tri-plane (EG3D) or on multiple isosurfaces (GRAM). This allows for much faster synthesis, compared to π -GAN, while at the same time improving visual quality. Further extensions of EG3D are PanoHead [2], which synthesizes 360° heads using additional training data, or WYSIWYG [38], allowing for high resolution rendering without super resolution.

3DGS GANs 3D Gaussian Splatting [21] represents a 3D scene with a set of 3D Gaussian primitives, where every primitive is described by its position, color, covariance, and opacity. Leveraging differentiable rendering, 3DGS optimizes the attributes of all Gaussian primitives to improve over a loss function.

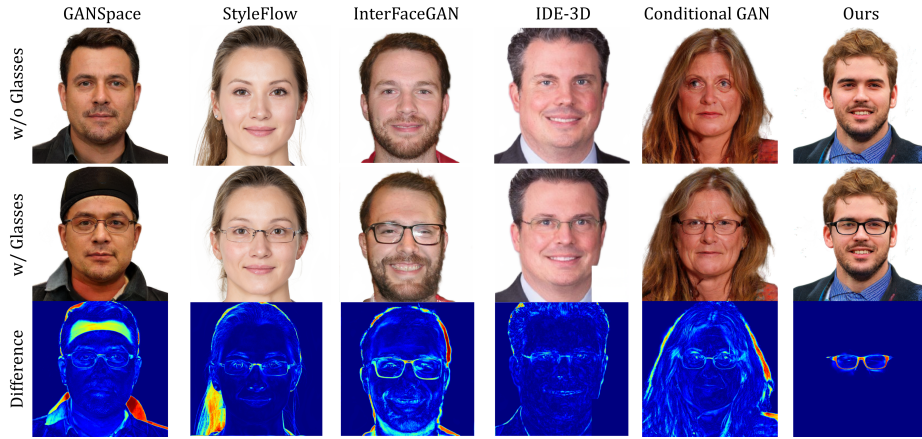


Fig. 2: Comparison between GAN editing methods for adding glasses. Existing methods exhibit slight changes, whereas our method cannot change the appearance as the glasses generation is handled in an independent generator branch.

This loss can be an image loss when reconstructing a 3D scene from multiple input views, or an adversarial loss when applying 3DGS in a 3D GAN framework. In contrast to NeRFs [30], 3DGS scenes are explicit and provide very efficient rendering. This allows for real-time rendering and simple integration into VR environments, video games, or even mobile applications. With all these advantages, multiple works have proposed 3DGS based GANs that replace the implicit NeRF representation with an explicit 3DGS scene [3, 4, 17, 26, 26, 43] .

A key distinction among the existing 3DGS GAN methods is the initialization of the position. On the one hand, GGHead [22], GAIA [43], and EGG3D [26] synthesize UV maps with attributes and position offsets of Gaussians on a template mesh derived from FLAME [27]. This template-based formulation produces high quality visual results and even enables animation control with GAIA [43] and EGG3D [26]. On the other hand, approaches like GSGAN [17] and CGS-GAN [4] model Gaussian positions as an unconstrained point cloud, optimized by a transformer architecture inspired by point-cloud generative models [14, 32, 46]. To impose structural coherence within the unconstrained point cloud representation, both GSGAN and CGS-GAN leverage a hierarchical structure of Gaussian primitives in which smaller Gaussian primitives are positioned relative to larger ones. As a result, both methods produce state-of-the-art results without relying on a predefined head template. Further, CGS-GAN [4] eliminates explicit view conditioning, which could lead to representations that appeared realistic primarily from specific viewing directions in previous 3DGS GANs. By eliminating this dependency, CGS-GAN encourages the generator to learn a view-consistent 3D structure, thereby improving multi-view coherence. This is demonstrated by the FID_{3D} metric that evaluates the 3D generator without informing the model in advance from which viewing direction the 3D model will be rendered.

Our approach builds upon CGS-GAN, as it achieves strong multi-view 3D consistency among existing unconstrained 3DGS GANs and does not rely on predefined template meshes. This is particularly important, since we do not assume the availability of template geometries for hair, glasses, face, or torso.

Latent Space Editing Editing is often described as a core strength of GANs compared to multi-view reconstruction methods or diffusion models. However, achieving strictly disentangled editing control over individual attributes remains challenging. As GANs are trained to approximate the full data distribution, the unconditional latent space captures dataset specific correlations and biases. For human head generation, the latent space strongly entangles hair and skin color, making it difficult to change one without the other. To counteract this, several methods aim to separate the latent space after training into regions that correlate with interpretable image features. GANSpace [18] computes PCA components of the latent space and heuristically assigns the components to face features. Typically, the first components affect major modifications, such as the identity, while minor components adjust subtle features, like lighting or expression. Although this strategy enables intuitive edits without external guidance, it provides only limited control and strong attribute entanglement. InterFaceGAN [34] extends the idea by identifying attribute-specific editing directions in the latent space using a pre-trained classifier. Similarly, StyleFlow [1] uses a pre-trained classifier to remap the entangled latent space to a learned conditional latent space. Although classifier-based editing methods generally improve attribute controllability, they remain constrained by the underlying entangled representation. An example of this strong entanglement is shown in Fig. 2. Here, we observe slight changes in identity or hairstyle when adding glasses.

Reference based Editing Reference based editing methods aim to combine attributes of two input images. For example, Barbershop [47] and HairCLIPv2 [39] transfer the hairstyle of a reference person onto a target identity. Similarly, StyleFusion [19] or TriplaneEdit [5] fuse features of two inputs and enable precise feature swapping guided by pre-trained masking networks. These methods achieve improved disentanglement compared to latent space editing, however, do not allow free exploration of the latent space, given the requirement of a reference image as input. Further, unlike latent space editing methods that operate in real-time, these approaches often take several minutes [47] to compute a single output image.

Compositional Generation The idea of decomposing the generation into multiple generators has been explored in prior works. For example, GIRAFFE [41], GSGAN [17] and EGG3D [26] split the generation of foreground and background using separate generators. In addition, separating hair and face generation was achieved in 3DGH [15] and EGG3D [26]. In contrast to our approach, however, existing methods either constrain the separation with explicit geometric priors,

such as positioning skin Gaussians on the surface of a FLAME head [26], or using a segmentation mask during training to guide the separation [15].

Similarly to compositional generation, several approaches [9, 35, 36, 44, 48] learn a direct mapping between 2D segmentation masks and generated images. This allows drawing or editing the outline of face regions, like hair, nose, or mouth, which then results in a realistic representation with same proportions.

Among existing GAN editing methods, our approach is mostly related to latent editing methods. Instead of shaping the proportions with segmentation masks or fusing attributes of reference images, our method splits the generation process semantically, allowing full exploration of the latent space of each component in real-time. For this reason, we mainly compare the editing quality with latent space methods and conventional conditional GANs.

Distinction to Diffusion Models Recent diffusion models [12, 24] for image and video synthesis have surpassed the quality of 2D GANs [6, 20, 33]. Nevertheless, in the context of 3D generation, GANs remain particularly attractive, as they can learn 3D representations directly from 2D image collections without 3D ground truth. Diffusion models, in contrast, inherently require a 3D ground truth for the noising and denoising step during training. Although several works have applied diffusion models in 3D head avatar generation [10, 25], they either train on synthetic 3D data (generated by 3D GANs [25]), or use an intermediate model [28, 29] to convert 2D inputs into 3D pseudo ground truth. Further, GANs offer a smoother latent space [45] and allow for real-time synthesis.

3 Method

To address the limitations of latent editing methods and entangled conditional GANs, we propose a compositional 3D GAN architecture that enables component-level control, using sparse conditions like colors or boolean flags. Our method consists of two main ideas. Firstly, a compositional GAN architecture to produce disentangled 3D components. Secondly, several techniques to improve the quality and consistency when mixing different components during inference.

3.1 Compositional GAN Architecture

To enable independent manipulation of semantic attributes, we construct the generator as a composition of multiple sub-generators. As shown in Fig. 3, we allocate a dedicated 3DGS generator, similar to CGS-GAN [4], to each semantic region, i.e., hair, face, glasses and torso. The complete 3D head is then formed by composing the Gaussian primitives into a single coherent scene representation. The composed 3D scene is then rendered into a 2D image using a differentiable 3DGS renderer, which is forwarded to the discriminator. To enforce component-wise specialization, we supply region-specific attribute annotations only to the corresponding sub-generator. Here, we differentiate between additive components like glasses, which are turned on and off, and permanent components like the skin, which are present in all head models.

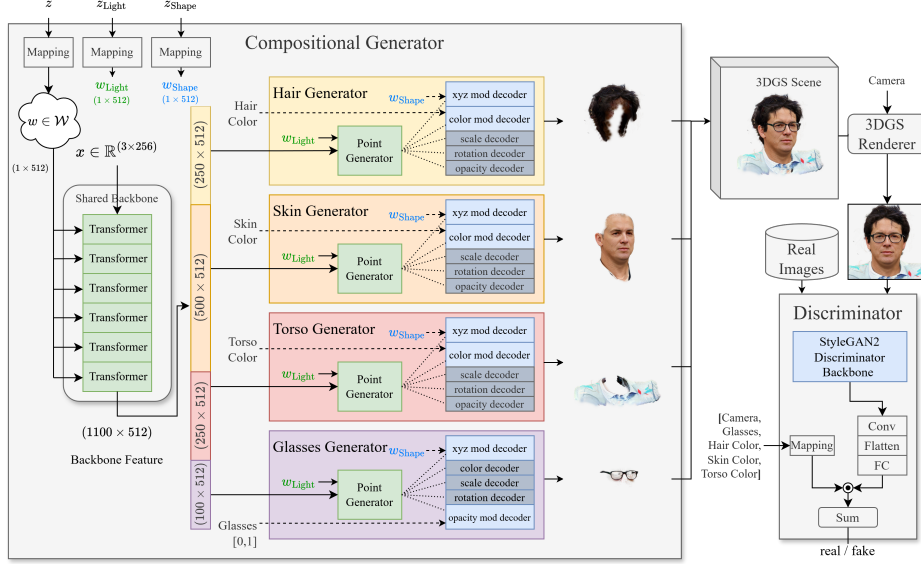


Fig. 3: Overview of our compositional 3D GAN architecture. Initially, we generate shared features with a transformer backbone that we partition and distribute to the sub-generators. Each sub-generator then produces a 3DGS scene using an adapted CGS-GAN [4] generator. In the final decoder layer, we modulate the colors given the conditioning from the dataset. Additionally we provide limited shared information w_{Shape} and w_{Light} to allow coordination along shape and lighting.

Additive Components To describe components, which are not always present during generation (like glasses), we use a boolean flag that describes the presence or absence of this component. If the component should be present, we generate it and add it to the composition, while we hide it by subtracting a large value from the Gaussian opacity when it is absent. At the same time, we provide the boolean flag as 0 or 1 to the discriminator, which penalizes the generator if the respective component is incorrectly missing or appearing. As the boolean flag is exclusively provided to the respective sub-generator, it implicitly learns to become responsible for generating this component. Similarly, if the sub-generator starts to generate more than necessary, for example, if the glasses generator produces the eyes or the skin, those features will be missing from the composition, when the output is hidden. As a result, the generator learns to precisely limit the generation to the designated component.

Permanent Components To synthesize components always present during generation (like the skin), we have to define a description that exclusively relates to the respective region. For our purpose, we select the color as a suitable description to separate skin, hair, and the torso. Additionally, this gives us the ability to control the color next to the overall latent vector. Specifically, we use

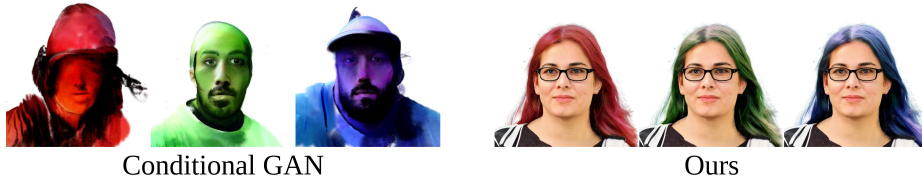


Fig. 4: Failure example of conditional GANs when providing colors that have never been seen in training. With conditional GANs, such inputs push the latent code into unexplored regions resulting in drastic image changes. Our method directly converts the input into Gaussian color attributes, leaving the geometry unchanged.

a sparse 10-bin RGB histogram aggregated over the respective head region. This color histogram is then applied to modulate the color attribute of the Gaussian primitives inside the respective sub-generator. This is done in the *color mod decoder* layer, shown in Figure 3, which converts the features of the point generator into Gaussian color attributes. Similar to the additive component, we provide the same color information to the conditional discriminator, which then produces a loss if the target color is not met. This sparse information is sufficient to cleanly separate the components of our 3D human head into hair, skin and torso. By only modulating 3DGS color attributes, we achieve very good generalization for novel colors. As shown in Figure 4, our generator learns a direct mapping from color to 3DGS features, whereas naive conditional GANs that condition the latent vector on an input conditioning usually break when provided with out-of-distribution conditions.

Discriminator We employ the conditional discriminator architecture of EG3D [7] with minimal changes. In addition to the camera conditioning, we concatenate all color information and the boolean glasses flag into one single vector and multiply it to the discriminator features. In contrast to existing compositional GANs [15, 26], our discriminator receives the entire image instead of a masked region, as the semantic separation is entirely handled within the generator. We consider this design advantageous, as segmentation masks are often not precise, while additionally mask-based supervision is difficult to apply during early training stages where the generator produces less structured outputs.

3.2 Global Consistency

With the architecture described in 3.1, the generator is now able to separately synthesize hair, glasses, face, and torso. However, with this design, the components will not fit well together, when synthesizing novel combinations with different latents in each sub-generator during inference. To enforce better compatibility, we propose several techniques that improve global consistency in shape and light, while also preparing the generator for novel component combinations already during training.



Fig. 5: Demonstration of different shape context vectors (left), light context vectors (middle) and the EMA face geometry (right).

Latent Mixing The most straightforward way to enforce better compatibility during inference is by already producing mixed components during training. This is achieved by randomly swapping the backbone features of one component during the forward pass in the generator. This process essentially simulates the editing of a generated 3D head during inference. We then combine the mixed 3DGS scenes into one scene by concatenating the 3DGS features and forward the renderings to the discriminator. The discriminator then produces a training signal that penalizes the generator if the components do not fit together. This avoids generating gaps or overlaps between mixed components.

Shape Context With the latent mixing method introduced in 3.2, we enforce better compatibility between components that were synthesized from different latent vectors. This, however, poses the challenge that in the training dataset, not all hairstyles, glasses or torso can possibly fit together. For example, the hair of a young child with a small head will likely not fit on a very large head. To avoid that the generator collapses to a single head shape that always fits together, we introduce a small latent code ($w_{\text{Shape}} \in \mathbb{R}^{512}$ in Fig. 3) that we share across all sub-generators and is never mixed during latent mixing. To make sure that the sub-generators do not use this shared signal to communicate anything but the shape, we forward this latent code only to the final xyz decoder layer in each sub-generator. This way, all other 3DGS attributes cannot be changed. Further, as the final position decoder is constructed as a small MLP with limited capacity, the model is prevented from encoding complex identity changes through this latent code. Fig. 5 demonstrates the effect of only changing the shape context vector, resulting in different head shapes but consistent identity.

Light Context Similarly to inconsistencies in shapes and sizes, all sub-generators have to agree on a shared global lighting condition. Otherwise, each sub-generator may synthesize different directional lighting, resulting in visually implausible compositions. Therefore, we provide the sub-generators with a shared low dimensional lighting latent code (see $w_{\text{Light}} \in \mathbb{R}^{512}$ in Fig. 3) that is never mixed during latent mixing. Intuitively, to achieve this, we would simply modulate the

output colors in the decoder layers, similarly to the shape context vector. However, as the final color decoder layer provides very limited capacity, it cannot model complex shadows or reflections. For this reason, we condition the input latent vector with the light context vector. As this allows the generator to modify everything, and not just the lighting, we then apply a regularization that penalizes the generator if two renderings from the same latent, but with different light context, differ too much. Here, we use the LPIPS distance of two images, rendered from the same camera pose. If the regularization strength is set correctly, we find that the generator implicitly learns to use this shared vector exclusively for lighting changes, as demonstrated in Fig. 5.

Exponential Moving Average Regularization In a compositional generator setting, one component may under-represent regions that are statistically also occupied by another component. In our case, the face generator may reduce coverage near the hairline, implicitly relying on the hair generator to complete the geometry. This leads to geometric inconsistencies when novel component combinations are synthesized. To avoid this, we introduce a geometric consistency loss penalizing deviations of the generated face geometry from an exponential moving average (EMA) face geometry. Specifically, we track the xyz coordinates and the opacity values, which are always in the same order. As the EMA face captures the entire forehead (Fig. 5 right), given that the majority of faces have a visible forehead, this loss pushes the generator towards synthesizing a more complete face.

4 Experiments

We compare our results from two main points of view. First, we evaluate the overall visual quality, compared to recent unconditional 3DGS GANs, and second, we compare the editing performance to existing GAN editing methods. Specifically, we measure the disentanglement, while also evaluating how well the output image represents the target label. Finally, we apply several ablations and perform a latent space analysis with GANSpace [18] on the sub-latent spaces of each component. Further examples and training details are provided in the supplementary materials.

To compare the visual quality and editing performance with a conventional conditional GAN [20, 31], we provide an additional baseline, in which we train the most recent 3DGS GAN (CGS-GAN [4]) with the same color and glasses conditioning. Here, however, the conditioning is directly forwarded as a single vector to the generator mapping network, while using the same discriminator.

Dataset We use FFHQC [4], which is a subset of FFHQ [20] with less occluding objects and larger cropping. To extract color information for the hair, skin and toros, we use BiSeNet [42] to extract rough regions and compute a 10-bin histogram on each color channel. We then concatenate all three histograms into one

30-dimensional vector that serves as the color conditioning for our compositional training. As we do not need precise segmentation masks, we shrink the respective regions, to avoid leaking color information to adjacent regions. Finally, we obtain the binary glasses flag from the FFHQ-features repository [11].

Metrics We use the Fréchet Inception Distance (FID) [16] to estimate the visual quality of the generated images. FID computes the difference between Inception-V3 [37] features for synthesized and real images. A low value indicates a very similar distribution, suggesting good visual quality. In particular, to measure the quality of the 3D heads, we employ the $\text{FID}_{3\text{D}}$ metric, which evaluates FID without using camera view conditioning to a generator. Essentially, $\text{FID}_{3\text{D}}$ hides the camera pose from the generator during the synthesis, which better reflects the actual 3D quality.

To evaluate editing disentanglement, we measure the extent to which editing one attribute affects unrelated regions. For this evaluation, we use the face parsing network BiSeNet [42] to obtain the regions that should not change during editing. On those patches, we then compute the LPIPS distance before and after editing. A low value denotes low attribute entanglement. In addition, we compute a recall for how often we see glasses when the glasses label was activated.

Results on Visual Quality In Tab. 1 we compare the $\text{FID}_{3\text{D}}$ metric with existing unconditional 3DGS GANs, as well as our conditional GAN baseline. Although the training is much more challenging, given that our generator has to coordinate the separation across different sub-generators, we achieve state-of-the-art $\text{FID}_{3\text{D}}$ results with 4.42 on 512^2 resolution and 5.43 on 1024^2 resolution. As our method synthesizes no backgrounds and FID is sensitive to background [23], we cannot directly compare it to EGG3D [26] (FID 7.51 with FFHQ 512 resolution with background). A visual comparison is shown in Fig. 6. Like CGS-GAN [4], our model produces high quality renderings with consistent side view performance. In contrast, EGG3D [26], GSGAN [17] and GGHead [22] produce lower quality towards steep viewing angles, as they condition the 3D synthesis on a specific viewing direction.

Table 1: Unconditional image quality measured with $\text{FID}_{3\text{D}}$ across recent 3DGS GAN methods using the FFQHC dataset at 512 and 1024 resolution.

$\text{FID}_{3\text{D}}$	$\downarrow 512^2$	$\downarrow 1024^2$
GSGAN [17]	7.68	-
GGHead [22]	7.78	14.27
CGS-GAN [4]	4.53	5.25
Cond CGS-GAN	4.52	-
Ours	4.42	5.43



Fig. 6: Visual comparison to recent 3DGS GANs at 512^2 resolution with truncation $\psi = 0.8$. EGG3D [26], GSGAN [17] and GGHead [22] produce worse side views, as those methods condition the generation on the viewing direction, which we set to a fixed frontal view. For EGG3D [26], we visualize their results without background using their official checkpoint that is trained with FFHQ with background.

Results on Editing Quality The strong side of our method lies in the editing performance compared to existing real-time GAN editing methods. As our model generates different components with separate sub-generators, we allow changing one attribute without touching the others. This is demonstrated in Fig. 1, 2, and 7, while we also supported our results numerically in Tab. 2. Here, we add glasses or change the hairstyle using the respective editing method and measure the LPIPS distance of before and after editing in unrelated regions. We observe significant perceptual changes, especially with GANSpace [18] or with the conditional GAN baseline. Our approach in contrast produces very little changes. The value is not 0 as the model produces small shadows under the glasses which are measured as slight perceptual differences.

In Tab. 3, we measure how often the editing method actually produces the target label. Specifically we select 1000 images without glasses and apply the respective editing method to add glasses. Here, GANSpace [18] and InterFaceGAN [34] show the least reliable results, as they only show glasses in about 50% or 60%. Although a higher recall could be gained from those methods, it would come with a tradeoff in LPIPS stability, which we calculate in Tab. 2 with the same image pairs. StyleFlow [1] and our conditional GAN baseline show much



Fig. 7: Traversing along the first PCA direction of each generator. Next to overall identity changes in the face, we observe a semantic separation between large sunglasses and small reading glasses, long and short hair, and formal cloth and an open jacket.

stronger results by achieving a recall of about 95%. Our method achieves almost 100% recall, as the model simply turns on the respective glasses sub-generator when the glasses label becomes active.

Sub-latent Space Analysis To gain a better intuition on how the latent space of each sub-generator behaves, we apply the GANSpace [18] approach on top of our method. Specifically, we compute the PCA components for each latent space and perform linear interpolation along the main directions. The result is demonstrated in Fig. 7. Here, we observe that each latent space encodes semantic features along the main PCA components. When changing the face latent code, we observe a shift from masculine to feminine appearance. Further, the glasses latent space separates large sunglasses and small reading glasses along the main PCA component. Similarly, the hair is changed from long hair to short hair, and the torso shifts from an open jacket to a closed suit.

Table 2: We measure the LPIPS distance in unrelated regions after adding glasses or changing the hairstyle.

LPIPS Stability	Glasses ↓	Hair ↓
GANSpace [18]	0.254	0.165
InterFaceGAN [34]	0.196	-
StyleFlow [1]	0.172	0.133
Cond CGS-GAN	0.239	0.149
Ours	0.016	0.014

Table 3: Recall on how often glasses are present when the glasses label is active.

Editing Recall	Glasses ↑
GANSpace [18]	0.508
InterFaceGAN [34]	0.632
StyleFlow [1]	0.948
Cond CGS-GAN	0.952
Ours	0.997

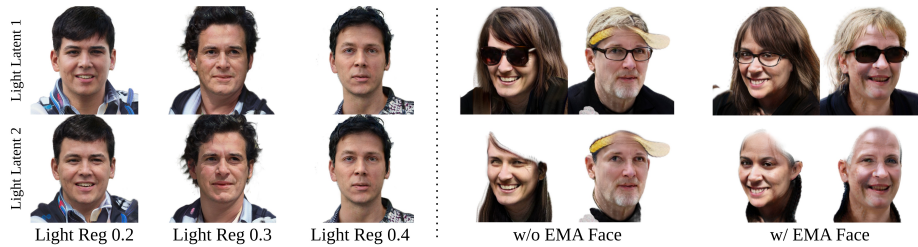


Fig. 8: Left: Ablation of different light context regularization strengths, with 0.3 showing a good tradeoff between too much and too little changes. Right: Ablation without the EMA face regularization, which produces incomplete foreheads (the images are taken from an early training checkpoint).

Light Context Ablation The strength of the light context regularization determines how perceptually different we allow the components to be when changing only the shared lighting context vector. Here, our target is to set the strength to an appropriate value so that large changes in hairstyle or identity are eliminated, while subtle changes like light and shadows are allowed. In a series of experiments we find that an optimal tradeoff is found when setting the value in a way that the mean LPIPS distance is about 0.3. This is shown in Fig. 8 (left) where we observe too large changes when the regularization weight is set too small, and no changes at all if regularization weight is too high. Although this parameter is very sensitive, it does not affect the overall training stability.

EMA Geometry Loss Ablation Similarly to the shape context, we encourage the generator to produce matching components by regularizing the distance between the generated face geometry and an EMA face geometry. As demonstrated in Fig. 8 (right), this avoids generating incomplete face geometries that stop just below the matching hairline, while also preventing the head generator from synthesizing outliers like headward.

Latent Mixing Ablation By synthesizing mixed examples during training, we ensure better compatibility between components during inference. In Tab. 4, we test different mixing probabilities. A too high mixing probability harms the overall training, resulting in lower FID, while a too little mixing produces poor visual results when editing during inference. To measure this, we calculate another FID score, where we only evaluate the quality of mixed compositions. For our final method we select 10% as it yields a good tradeoff between FID and FID_{Mix} .

Table 4: FID and FID_{Mix} results with different latent mixing probabilities.

Mixing Prob.	FID ↓	FID_{Mix} ↓
0%	4.88	10.51
10% (Ours)	4.42	8.32
20%	4.98	9.07
30%	5.16	8.49
40%	5.97	6.53

5 Conclusion

We present a compositional 3D GAN architecture that decomposes the 3D human head generation into semantically localized components. With our design that avoids explicit geometric priors or segmentation masks, the generator implicitly learns a clean separation from colors or boolean flags. This allows for precise GAN editing of hairstyle or glasses without modifying the rest. By allowing the sub-generators minimal communication using shared context latent codes, we improve the generalization while achieving manual control over lighting and shape without prior annotation. Further, we apply two training regularizations that improve the visual quality when composing a 3D head from different latent codes. In our experiments, we achieve competitive visual quality measured with FID, while achieving low entanglement and high editing accuracy.

In future work, the compositional framework could be extended to additional components, such as hats, beards, or accessories. Furthermore, none of the components in our generator architecture are specifically designed for 3D human heads. Hence, it can be adapted to other domains where objects can be decomposed into meaningful parts. This can include full body avatars with disentangles body and cloth for virtual try on, or 3D cars with customizable parts. Such extensions allow the method to be deployed for synthesizing annotated 3D simulation environments with a wide variety.

Acknowledgements

This work has partly been funded by the German Research Foundation (project 3DIL, grant no. 502864329) and the European Commission (Horizon Europe project Luminous, grant no. 101135724).

References

1. Abdal, R., Zhu, P., Mitra, N.J., Wonka, P.: Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Transactions on Graphics (ToG)* **40**(3), 1–21 (2021)
2. An, S., Xu, H., Shi, Y., Song, G., Ogras, U.Y., Luo, L.: Panohead: Geometry-aware 3d full-head synthesis in 360deg. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. pp. 20950–20959 (2023)
3. Barthel, F., Beckmann, A., Morgenstern, W., Hilsmann, A., Eisert, P.: Gaussian splatting decoder for 3d-aware generative adversarial networks. In: *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. pp. 7963–7972 (2024). <https://doi.org/10.1109/CVPRW63382.2024.00794>
4. Barthel, F., Morgenstern, W., Hinzer, P., Hilsmann, A., Eisert, P.: Cgs-gan: 3d consistent gaussian splatting gans for high resolution human head synthesis. In: Belgrave, D., Zhang, C., Lin, H., Pascanu, R., Koniusz, P., Ghassemi, M., Chen, N. (eds.) *Advances in Neural Information Processing Systems*. vol. 38, pp. 89230–89258. Curran Associates, Inc. (2025), <https://proceedings.neurips.cc/>

- `paper_files/paper/2025/file/80b7bec60081f95d900973509744a306-Paper-Conference.pdf`
5. Bilecen, B.B., Yalin, Y., Yu, N., Dundar, A.: Reference-based 3d-aware image editing with triplanes. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 5904–5915 (2025)
 6. Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training for high fidelity natural image synthesis. In: International Conference on Learning Representations (2019), <https://openreview.net/forum?id=B1xsqj09Fm>
 7. Chan, E.R., Lin, C.Z., Chan, M.A., Nagano, K., Pan, B., De Mello, S., Gallo, O., Guibas, L.J., Tremblay, J., Khamis, S., et al.: Efficient geometry-aware 3d generative adversarial networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 16123–16133 (2022)
 8. Chan, E.R., Monteiro, M., Kellnhofer, P., Wu, J., Wetzstein, G.: 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. pp. 5799–5809 (2021)
 9. Chen, A., Liu, R., Xie, L., Chen, Z., Su, H., Yu, J.: Sofgan: A portrait image generator with dynamic styling. *ACM Transactions on Graphics (TOG)* **41**(1), 1–26 (2022)
 10. Chen, X., Mihajlovic, M., Wang, S., Prokudin, S., Tang, S.: Morphable diffusion: 3d-consistent diffusion for single-image avatar creation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2024)
 11. Dcgm: Dcgm/ffhq-features-dataset: Gender, age, and emotion for flickr-faces-hq dataset (ffhq), <https://github.com/DCGM/ffhq-features-dataset>, accessed: 2026-05-14
 12. Gong, L., Hou, X., Li, F., Li, L., Lian, X., Liu, F., Liu, L., Liu, W., Lu, W., Shi, Y., et al.: Seedream 2.0: A native chinese-english bilingual image generation foundation model. *arXiv preprint arXiv:2503.07703* (2025)
 13. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. *Advances in neural information processing systems* **27** (2014)
 14. Guo, M.H., Cai, J.X., Liu, Z.N., Mu, T.J., Martin, R.R., Hu, S.M.: Pct: Point cloud transformer. *Computational visual media* **7**, 187–199 (2021)
 15. He, C., Li, J., Kirschstein, T., Sevastopolsky, A., Saito, S., Tan, Q., Romero, J., Cao, C., Rushmeier, H., Nam, G.: 3dgh: 3d head generation with composable hair and face. *ACM Transactions on Graphics* **44**(4), 1–12 (2025)
 16. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems* **30** (2017)
 17. Hyun, S., Heo, J.P.: GSGAN: Adversarial learning for hierarchical generation of 3d gaussian splats. In: The Thirty-eighth Annual Conference on Neural Information Processing Systems (2024), <https://openreview.net/forum?id=sFaFDcVNbW>
 18. Härkönen, E., Hertzmann, A., Lehtinen, J., Paris, S.: Ganspace: Discovering interpretable gan controls. In: *Proc. NeurIPS* (2020)
 19. Kafri, O., Patashnik, O., Alaluf, Y., Cohen-Or, D.: Stylefusion: A generative model for disentangling spatial segments (2021)
 20. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: *Proc. CVPR* (2020)
 21. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics* **42**(4) (July 2023), <https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/>

22. Kirschstein, T., Giebenhain, S., Tang, J., Georgopoulos, M., Niekner, M.: GGHead: Fast and Generalizable 3D Gaussian Heads. In: SIGGRAPH Asia 2024 Conference Papers. SA '24, Association for Computing Machinery, New York, NY, USA (2024). <https://doi.org/10.1145/3680528.3687686>, <https://doi.org/10.1145/3680528.3687686>
23. Kynkäänniemi, T., Karras, T., Aittala, M., Aila, T., Lehtinen, J.: The role of imagenet classes in fr chet inception distance. In: ICLR (2023)
24. Labs, B.F., Batifol, S., Blattmann, A., Boesel, F., Consul, S., Diagne, C., Dockhorn, T., English, J., English, Z., Esser, P., Kulal, S., Lacey, K., Levi, Y., Li, C., Lorenz, D., M ller, J., Podell, D., Rombach, R., Saini, H., Sauer, A., Smith, L.: Flux.1 kontext: Flow matching for in-context image generation and editing in latent space (2025), <https://arxiv.org/abs/2506.15742>
25. Lan, Y., Tan, F., Qiu, D., Xu, Q., Genova, K., Huang, Z., Fanello, S., Pandey, R., Funkhouser, T., Loy, C.C., Zhang, Y.: Gaussian3diff: 3d gaussian diffusion for 3d full head synthesis and editing. In: ECCV (2024)
26. Li, G., Yang, H., Men, Y., Huang, D., Li, W., Yang, R., Wang, Y.: Generating editable head avatars with 3d gaussian gans. In: ICASSP 2025 (2025)
27. Li, T., Bolkart, T., Black, M.J., Li, H., Romero, J.: 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), <https://doi.org/10.1145/3130800.3130813>
28. Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 9298–9309 (2023)
29. Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: Syncdreamer: Generating multiview-consistent images from a single-view image. In: International conference on learning representations. vol. 2024, pp. 27676–27697 (2024)
30. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
31. Mirza, M., Osindero, S.: Conditional generative adversarial nets. *CoRR abs/1411.1784* (2014), <http://arxiv.org/abs/1411.1784>
32. Nichol, A., Jun, H., Dhariwal, P., Mishkin, P., Chen, M.: Point-e: A system for generating 3d point clouds from complex prompts. *arXiv preprint arXiv:2212.08751* (2022)
33. Sauer, A., Schwarz, K., Geiger, A.: Stylegan-xl: Scaling stylegan to large diverse datasets. In: ACM SIGGRAPH 2022 conference proceedings. pp. 1–10 (2022)
34. Shen, Y., Yang, C., Tang, X., Zhou, B.: Interfacegan: Interpreting the disentangled face representation learned by gans. *IEEE transactions on pattern analysis and machine intelligence* **44**(4), 2004–2018 (2020)
35. Sun, J., Wang, X., Shi, Y., Wang, L., Wang, J., Liu, Y.: Ide-3d: Interactive disentangled editing for high-resolution 3d-aware portrait synthesis. *ACM Transactions on Graphics (ToG)* **41**(6), 1–10 (2022)
36. Sun, J., Wang, X., Zhang, Y., Li, X., Zhang, Q., Liu, Y., Wang, J.: Fenerf: Face editing in neural radiance fields. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 7672–7682 (2022)
37. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826 (2016)
38. Trevithick, A., Chan, M., Takikawa, T., Iqbal, U., De Mello, S., Chandraker, M., Ramamoorthi, R., Nagano, K.: What you see is what you gan: rendering every pixel

- for high-fidelity geometry in 3d gans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22765–22775 (2024)
39. Wei, T., Chen, D., Zhou, W., Liao, J., Zhang, W., Hua, G., Yu, N.: Hairclipv2: Unifying hair editing via proxy feature blending. Proceedings of the IEEE/CVF International Conference on Computer Vision (2023)
 40. Xiang, J., Yang, J., Deng, Y., Tong, X.: Gram-hd: 3d-consistent image generation at high resolution with generative radiance manifolds. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2195–2205 (2023)
 41. Xue, Y., Li, Y., Singh, K.K., Lee, Y.J.: Giraffe hd: A high-resolution 3d-aware generative model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18440–18449 (2022)
 42. Yu, C., Wang, J., Peng, C., Gao, C., Yu, G., Sang, N.: Bisenet: Bilateral segmentation network for real-time semantic segmentation. In: Proceedings of the European conference on computer vision (ECCV). pp. 325–341 (2018)
 43. Yu, Z., Li, T., Sun, J., Shapira, O., Park, S., Stengel, M., Chan, M., Li, X., Wang, W., Nagano, K., Mello, S.D.: GAIA: Generative animatable interactive avatars with expression-conditioned gaussians. In: ACM SIGGRAPH (2025)
 44. Yun, K., Kim, C., Shin, H., Noh, J.: Ffacenerf: few-shot face editing in neural radiance fields. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 10825–10835 (2025)
 45. Zhang, K., Zhou, Y., Xu, X., Dai, B., Pan, X.: Diffmorpher: Unleashing the capability of diffusion models for image morphing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 7912–7921 (2024)
 46. Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V.: Point transformer. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 16259–16268 (2021)
 47. Zhu, P., Abdal, R., Femiani, J., Wonka, P.: Barbershop: Gan-based image compositing using segmentation masks (2021)
 48. Zhu, P., Abdal, R., Qin, Y., Wonka, P.: Sean: Image synthesis with semantic region-adaptive normalization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 5104–5113 (2020)