

---

# MiraGe: Editable 2D Images using Gaussian Splatting

---

Joanna Waczyńska<sup>1,2</sup> Tomasz Szczepanik<sup>1</sup> Piotr Borycki<sup>1</sup> Sławomir Tadeja<sup>3</sup> Thomas Bohné<sup>3</sup>  
Przemysław Spurek<sup>1,4</sup>

## Abstract

Implicit Neural Representations (INRs) approximate discrete data through continuous functions and are commonly used for encoding 2D images. Traditional image-based INRs employ neural networks to map pixel coordinates to RGB values, capturing shapes, colors, and textures within the network’s weights. Recently, GaussianImage has been proposed as an alternative, using Gaussian functions instead of neural networks to achieve comparable quality and compression. Such a solution obtains a quality and compression ratio similar to classical INR models but does not allow image modification. In contrast, our work introduces a novel method, MiraGe, which uses mirror reflections to perceive 2D images in 3D space and employs flat-controlled Gaussians for precise 2D image editing. Our approach improves the rendering quality and allows realistic image modifications, including human-inspired perception of photos in the 3D world. Thanks to modeling images in 3D space, we obtain the illusion of 3D-based modification in 2D images. We also show that our Gaussian representation can be easily combined with a physics engine to produce physics-based modification of 2D images. Consequently, MiraGe allows for better quality than the standard approach and natural modification of 2D images.

## 1. Introduction

Recent research has increasingly emphasized human perception and the understanding of the world through this lens

<sup>1</sup>Jagiellonian University, Faculty of Mathematics and Computer Science <sup>2</sup>Doctoral School of Exact and Natural Sciences <sup>3</sup>University of Cambridge <sup>4</sup>IDEAS Research Institute. Correspondence to: Joanna Waczyńska <joanna.waczynska@doctoral.uj.edu.pl>, Przemysław Spurek <przemyslaw.spurek@uj.edu.pl>.

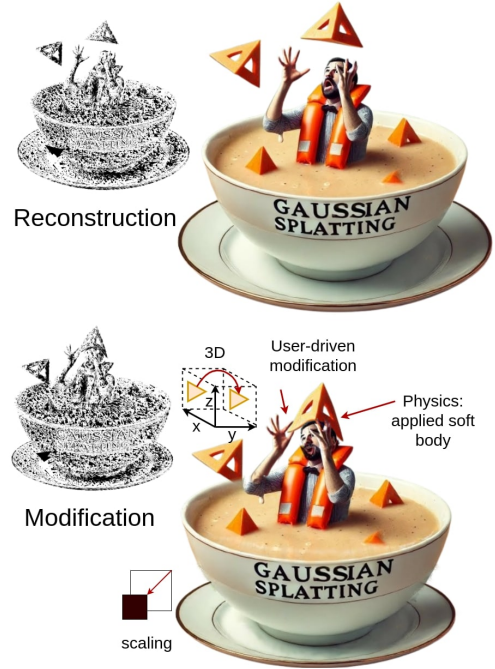


Figure 1. MiraGe encodes 2D images with parameterized Gaussians, enabling high-quality reconstruction and real-life-like modifications. The selected parts of the image can be transformed in 3D space, creating a 3D effect with a physics engine controlling movement and interactions.

(Lu, 2019; Davoodi et al., 2023). In line with this trend, we introduce a model that encodes 2D images by simulating human interpretation. Specifically, our model perceives a 2D image as a human would view a photograph or a sheet of paper, treating it as a flat object within a 3D space. This approach allows for intuitive and flexible image editing, capturing the nuances of human perception while enabling complex transformations (Fig. 1).

Gaussian Splatting (3DGS) framework models the structure of a 3D scene using Gaussian components (Kerbl et al., 2023). In the 2D domain, GaussianImage (Zhang et al., 2024) has shown promising results in image reconstruction by efficiently encoding images in the 2D space, with a strong focus on model efficiency and reduced training time. Unfortunately, GaussianImage does not support user-driven adjustments of scene objects, which is a key feature of

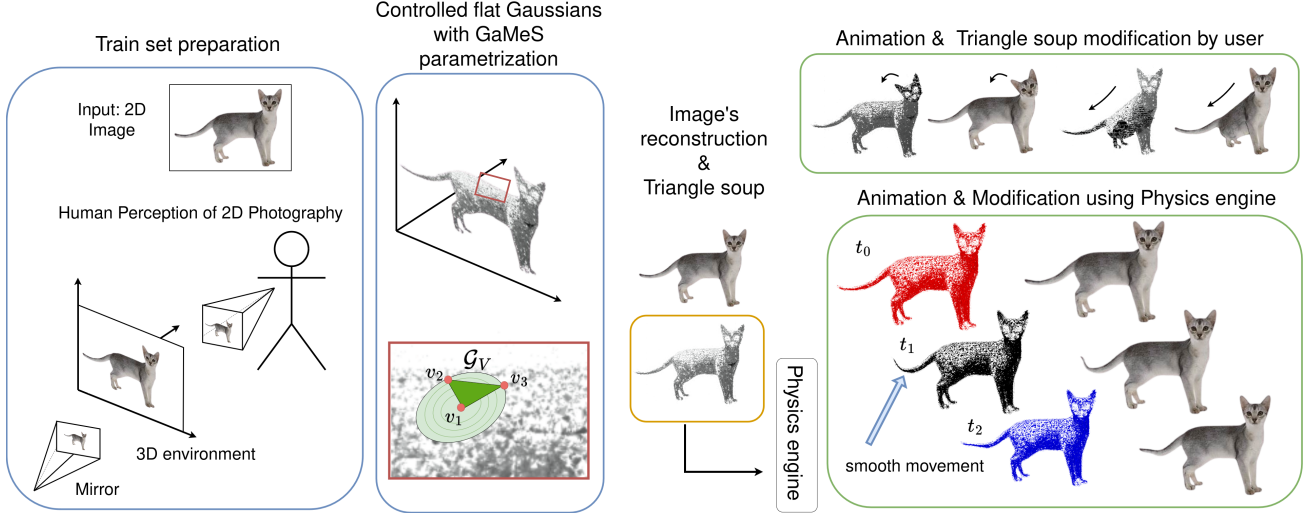


Figure 2. MiraGe employs 3D flat parameterized Gaussians in 3D space to encode 2D images, representing each flat Gaussian as three points, forming a cloud of triangles called a triangle soup. This representation enables real-time manipulation of the 3D triangle/point clouds, allowing for flexible, real-world modifications. The model seamlessly integrates with a physics engine, enhancing its applicability in dynamic environments.

3DGS. While GaussianImage has explored image representation using 2D Gaussians primarily for data compression, our research highlights an additional benefit, i.e., the use of parameterized flat 3D Gaussians for editing 2D images. In our work, we address this by introducing the MiraGe<sup>1</sup>, model, which encodes 2D images through the lens of human perception, bridging the gap between 2D image representation and 3D spatial understanding (Fig. 2).

Building on the foundational idea that humans intuitively can perform transformations on photographs—primarily through affine transformations and bending them beyond the 2D plane—we introduce a novel approach using flat Gaussians with GaMeS parametrization (Waczyńska et al., 2024). This capability enables our model to support image editing in both 2D and 3D spaces. Notably, our framework simplifies often difficult perspective adjustments by allowing intuitive modifications directly within the third dimension (Fig. 3).

In addition to classical edits, our model has the unique capability of interfacing with physics engines, enabling applications that enhance the realism and immersiveness of animations (Jiang et al., 2024). MiraGe treats the physics engine as a black box and offers three distinct methods for controlling Gaussians, i.e., 2D, Amorphous and Graphite. For 2D representation (2D-MiraGe) we used Taichi\_elements<sup>2</sup>, for 3D representation (Amorphous-MiraGe, Graphite-MiraGe)

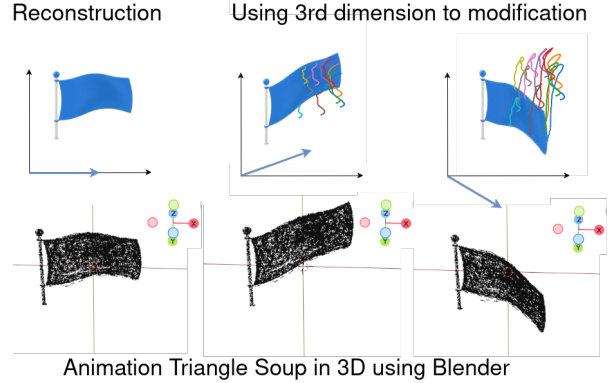


Figure 3. Parameterized flat 3D Gaussians provide a powerful representation of 2D images, enabling flexible editing in 3D space. Triangle Soup can be animated using tools like Blender. The colored lines depict the motion paths of 10 randomly selected points during the simulation.

we use Blender<sup>3</sup>. This flexibility makes our model highly applicable to various fields, such as computer graphics for populating spatial interfaces, where realistic, physics-factual 2D animations can be incorporated (Tadeja et al., 2023).

Embedding 2D images in 3D space allows for seamless integration of 2D and 3D objects, enabling the creation of dynamic backgrounds or interactive elements within animated scenes. This versatility extends to applications such as virtual reality, where 2D images can function as backdrops (Yin et al., 2024). This capability opens up new avenues for creative composition, offering a powerful toolset for

<sup>1</sup><https://waczjoan.github.io/MiraGe/>

<sup>2</sup>[https://github.com/taichi-dev/taichi\\_elements](https://github.com/taichi-dev/taichi_elements)

<sup>3</sup><https://www.blender.org> version 3.6

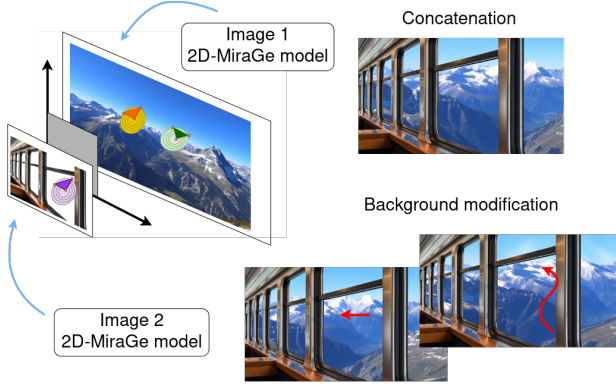


Figure 4. Two images were encoded using the MiraGe model on distinct planes within a 3D space. This setup allows for seamless integration of the encoded images, resulting in a collage-like composition. Moreover, the model facilitates editing capabilities, as illustrated here, with modifications to the background image (the rear plane).

users. The novelty of this work lies in its ability to enable easy, intuitive 3D transformations and integrations within a traditionally 2D framework, expanding the possibilities for both image editing and animation (Fig. 4).

Since high-quality image reconstruction is critical in animation, we compared MiraGe with other models, in particular with GaussianImage (Fig. 5), showing our state-of-the-art performance in the image reconstruction task.

It is worth highlighting that flat 3D Gaussians can be utilized for 2D images in four distinct scenarios, with modifications that emphasize how controlling the Gaussians during training affects the perspective of viewing each image (Fig. 6). The following constitutes a list of our key contributions:

- We introduce the MiraGe model, which represents 2D images using flat 3D Gaussian components, achieving state-of-the-art reconstruction quality.
- MiraGe enables the manipulation of 2D images within 3D space, creating the illusion of 3D effects.
- We integrate MiraGe with a physics engine, enabling physics-based modifications and interactions for both 2D and 3D environments.

## 2. Related Works

Our work builds on several key research areas, including image reconstruction techniques, Gaussian-based representations and Gaussian animation frameworks.

One rapidly growing area in image reconstruction is Implicit Neural Representations (INRs), which have attracted significant attention for their ability to model continuous signals, such as images, through neural networks (Klocek et al., 2019). INRs encode spatial coordinates and map them to corresponding values, such as RGB color, allowing

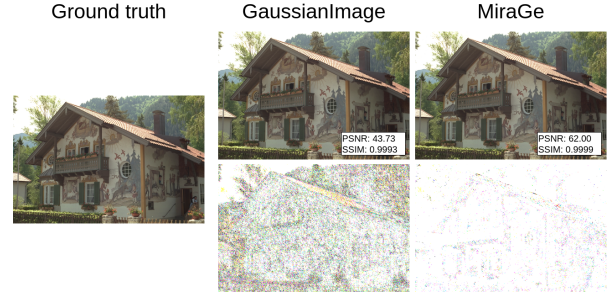


Figure 5. Visual comparison of two Gaussian-based methods for 2D image reconstruction. From left to right, the columns display the ground truth image, the GaussianImage reconstruction, and the MiraGe reconstruction. The bottom row illustrates the differences between the ground truth image and the results of each method.

for highly compact and efficient representations (Xie et al., 2022). This has led to the development of several specialized models for image INR, such as SIREN (Sitzmann et al., 2020a), Fourier feature mapping (Tancik et al., 2020), and WIRE (Saragadam et al., 2023a). The growing field of research resulted in further improvements of already existing solutions, e.g. in (Liu et al., 2024b), certain limitations of SIREN, namely the arising capacity-convergence gap, were successfully alleviated with the idea of variable-periodic activation functions. Yet another worth noting work from this area is (Müller et al., 2022) with INR solution designed to effectively perform on modern computer architecture utilizing a simple data structure concept of hashmap to offer speed-oriented image representation with high fidelity of the outcomes.

Alternative approaches to INRs were presented in GaussianImage (Zhang et al., 2024). Instead of neural networks, the authors propose to approximate 2D images using 2D Gaussian components. In practice, GaussianImage is a 2D version of 3DGS (Kerbl et al., 2023) that uses 2D Gaussians instead of their 3D version and a simplified rendering procedure. Thanks to such a modification, the GaussianImage is invariant to the order of Gaussian components. Therefore, such a model is numerically efficient.

GaussianImage represents each pixel color as a weighted sum of 2D Gaussians. The training procedure is similar to 3DGS without pruning. The authors show that such representation gives a similar reconstruction quality to classical INR models and is able to obtain a high compression ratio and fast rendering.

The interactive image editing of 2D images has been widely explored in computer graphics. Here, some methods leverage the current advancements in generative models. For instance, (Pan et al., 2023) introduce DragGAN, enabling point-based manipulation of images by performing them on the underlying manifold of GAN, achieving realistic edits. Similarly, (Shi et al., 2023) propose DragDiffusion, which



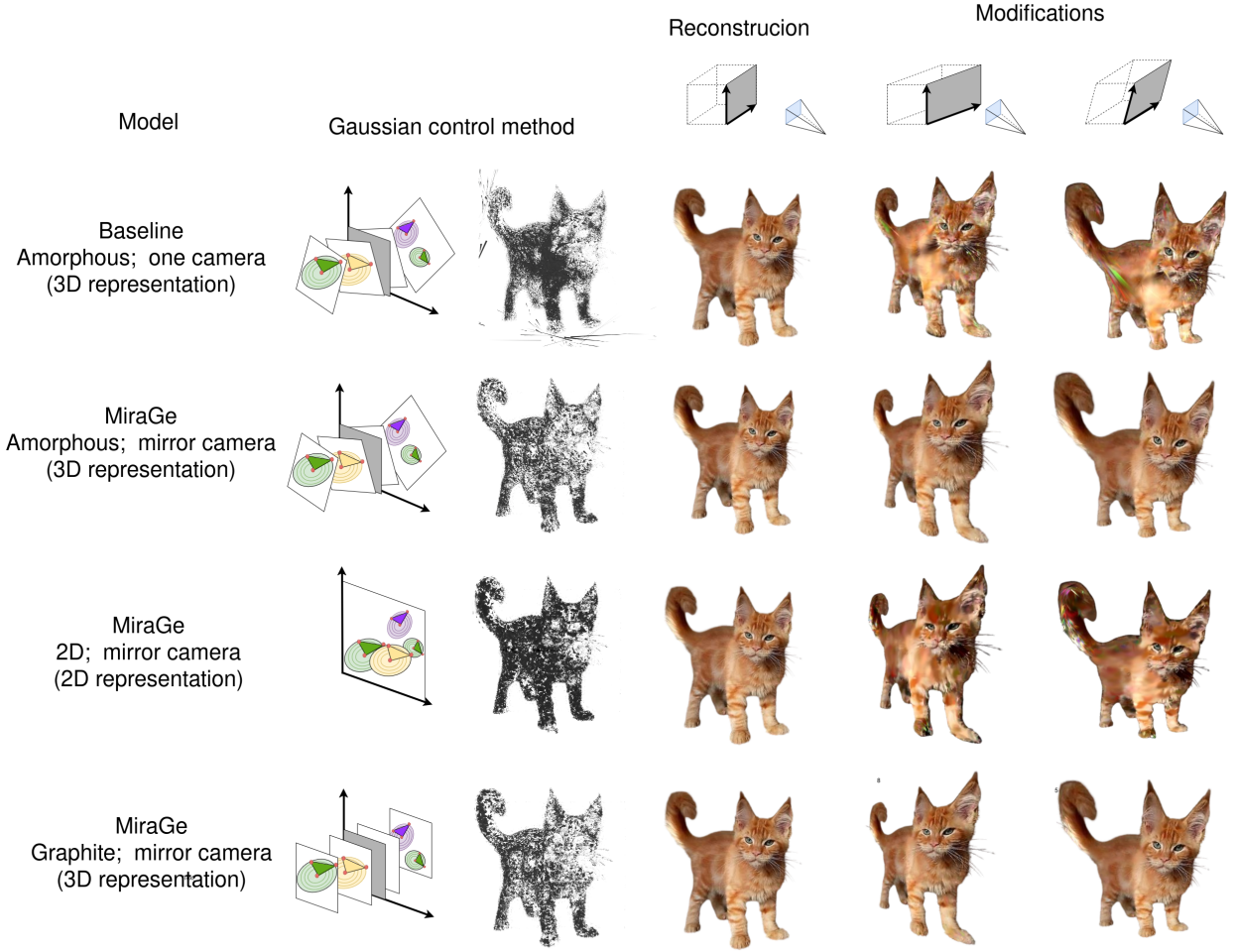


Figure 6. We demonstrate three approaches for Gaussian control: Amorphous, 2D, and Graphite. As a baseline, we utilize a single camera from the Amorphous setup. After applying perspective editing in 3D, the image shows noticeable deformation. In contrast, no deformation is observed when employing an additional camera. The model employs a mirror setup during training, with the Amorphous configuration achieving the best results for image reconstruction and 3D analysis. The 2D model represents images on a single plane. The Graphite model operates across multiple planes, making it ideal for 3D spatial reasoning and image combination.

extends the previous framework to diffusion models, enhancing the control and applicability of image editing. On the other hand, (Jacobson et al., 2011) propose bounded biharmonic weights for linear blending, which produce smooth and intuitive deformation for handles of arbitrary topology. (Wang et al., 2015) further advances this field by proposing linear subspace design, unifying linear blend skinning and generalized barycentric coordinates to provide a practical way of controlling deformations.

The representation and editing of objects using Gaussians is a well-explored topic in 3D graphics. In this field, meshes can be modified to simulate Gaussian editing (Guédon & Lepetit, 2024; Huang et al., 2024), or Gaussians can be directly parameterized and manipulated to achieve specific

outcomes (Waczyńska et al., 2024; Waczyńska et al., 2024). This approach enables flexible and continuous deformations, offering an intuitive method for controlling object shapes and rendering properties, which has proven particularly useful in tasks like texture mapping, surface smoothing, and dynamic simulations.

Gaussians enable precise and flexible editing of objects, providing continuous control over shapes and transformations. Moreover, integrating physics engines enhances these capabilities, allowing for more sophisticated and physically consistent modifications, such as simulating realistic interactions, deformations and movements in 3D environments. (Xie et al., 2024; Borycki et al., 2024).



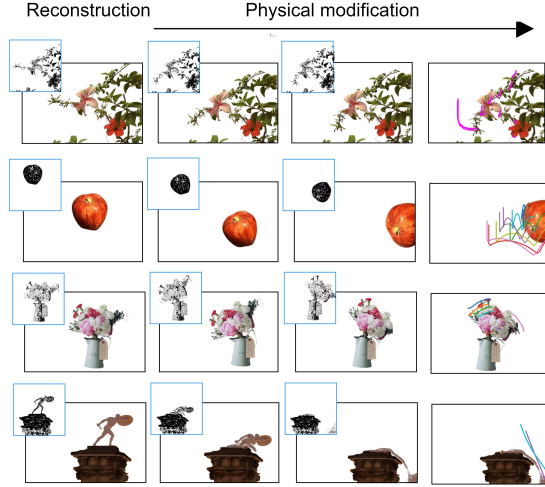


Figure 7. Integration of MiraGe with MPM enables realistic 2D image alterations. The first column shows the original image, the next two capture mid-simulation renders, and the last presents the final result with colored lines tracking point trajectories.

### 3. MiraGe: Editable 2D Images using Gaussian Splatting

Here, we describe in detail the inner workings of our MiraGe model. We start by presenting classical 3DGS. Next, we present GaMeS-based (Waczyńska et al., 2024) parametrization of flat Gaussians. In the end, we present our MiraGe and how it relates to prior works.

**3D Gaussian Splatting** 3DGS models 3D scene by a set of Gaussian components with color and opacity:

$$\mathcal{G} = \{(\mathcal{N}(\mathbf{m}_i, \Sigma_i), \sigma_i, c_i)\}_{i=1}^p,$$

defined by their mean (position)  $\mathbf{m}_i$ , covariance matrix  $\Sigma_i$ , opacity  $\sigma_i$ , and color  $c_i$ , which is represented using spherical harmonics (SH) (Fridovich-Keil et al., 2022).

During the rasterization stage, the 3DGS produces a sorted Gaussian list based on the projected depth information. Then, the  $\alpha$ -blending method is used to create the image. We refine the Gaussian parameters, color, and opacity in the training phase according to the reconstruction cost function. The optimal number of Gaussians required to represent a given object is not known a priori, and it is non-trivial to adjust the number of Gaussians. Hence, the initial number of Gaussians is a parameter of the method. The authors implement additional strategies for reducing and multiplying Gaussians. Gaussians with low opacity are removed, while those that change rapidly during optimization are multiplied. These strategies make the 3D Gaussian approach very efficient and capable of generating high-quality renders. We used this strategy to reconstruct 2D images, which distinguishes us from GaussianImage.

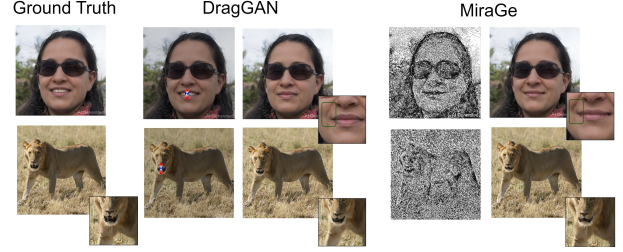


Figure 8. Visual comparison of image editing techniques, demonstrating the effectiveness of representing 2D images with parameterized Gaussians applied to Triangle Soup. This approach enables highly realistic animations, achieving results comparable to those of generative models. Specifically, local editing operations preserve fine details, such as a dimple on a face, without affecting unrelated regions. Moreover, we can achieve precise manipulations, including subtle edits like closing a lion’s mouth, underscoring the flexibility and control inherent in our method.

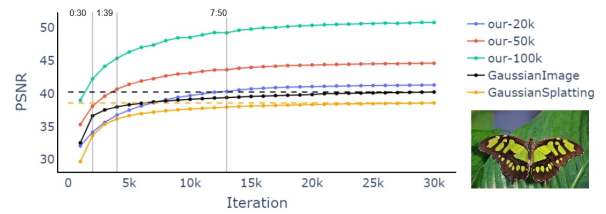


Figure 9. Comparison of PSNR obtained on a butterfly image from DIV2K dataset by MiraGe, GaussianImage and GS. Vertical lines represent iteration, where MiraGe obtained better results than GaussianImage, the time in min:sec format above each line is the training time until this iteration.

**GaMeS Parametrization of Gaussian Component** In MiraGe, we use flat Gaussian components in 3D space. In such a model we use Gaussian components with a covariance matrix  $\Sigma$ , factored as:  $\Sigma = RSSR^T$ , where  $R$  is the rotation matrix, and  $S$  is a diagonal matrix containing the scaling parameters. However, we force one of the scale parameters to be zero. Consequently, we obtain a collection of flat Gaussian:

$$\mathcal{G} = \{(\mathcal{N}(\mathbf{m}_i, R_i, S_i), \sigma_i, c_i)\}_{i=1}^p, \quad (1)$$

where  $S = \text{diag}(s_1, s_2, s_3)$ , with  $s_1 = \varepsilon$ , and  $R$  is the rotation matrix defined as  $R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$ , with  $\mathbf{r}_i \in \mathbb{R}^3$ . In such a case, we can use GaMeS parametrization to represent flat Gaussian by triangle-face mesh. This mapping is denoted by  $\mathcal{T}(\cdot)$ . When applied, this parametrization generates a set of triangles labeled as triangle soup.

To outline the GaMeS parametrization, consider a Gaussian component  $\mathcal{N}(\mathbf{m}, R, S)$ , characterized by the mean  $\mathbf{m}$ , the rotation matrix  $R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$  and the scaling matrix  $S = \text{diag}(\varepsilon, s_2, s_3)$ . Then its face representation  $\mathcal{N}(V)$  is based on a triangle:  $V = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3] = \mathcal{T}(\mathbf{m}, R, S)$

with the vertices defined as:  $\mathbf{v}_1 = \mathbf{m}$ ,  $\mathbf{v}_2 = \mathbf{m} + s_2 \mathbf{r}_2$ , and  $\mathbf{v}_3 = \mathbf{m} + s_3 \mathbf{r}_3$ . Conversely, given a face (triangle) representation  $V = [\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3]$ , we can recover the Gaussian component  $\mathcal{N}(\hat{\mathbf{m}}, \hat{R}, \hat{S}) = \mathcal{N}(\mathcal{T}^{-1}(V))$  through the mean  $\hat{\mathbf{m}}$ , the rotation matrix  $\hat{R} = [\hat{\mathbf{r}}_1, \hat{\mathbf{r}}_2, \hat{\mathbf{r}}_3]$ , and the scaling matrix  $\hat{S} = \text{diag}(\hat{s}_1, \hat{s}_2, \hat{s}_3)$ , where the parameters are defined by the following formulas:

$$\begin{aligned}\hat{\mathbf{m}} &= \mathbf{v}_1, \quad \hat{\mathbf{r}}_1 = \frac{(\mathbf{v}_2 - \mathbf{v}_1) \times (\mathbf{v}_3 - \mathbf{v}_1)}{\|(\mathbf{v}_2 - \mathbf{v}_1) \times (\mathbf{v}_3 - \mathbf{v}_1)\|}, \\ \hat{\mathbf{r}}_2 &= \frac{(\mathbf{v}_2 - \mathbf{v}_1)}{\|(\mathbf{v}_2 - \mathbf{v}_1)\|}, \quad \hat{\mathbf{r}}_3 = \text{orth}(\mathbf{v}_3 - \mathbf{v}_1; \mathbf{r}_1, \mathbf{r}_2), \\ s_1 &= \varepsilon, \quad \hat{s}_2 = \|\mathbf{v}_2 - \mathbf{v}_1\|, \quad \text{and} \quad \hat{s}_3 = \langle \mathbf{v}_3 - \mathbf{v}_1, \hat{\mathbf{r}}_3 \rangle.\end{aligned}$$

Here  $\text{orth}(\cdot)$  denotes a single step of the Gram-Schmidt process (Björck, 1994). Accordingly, the corresponding covariance matrix of a Gaussian distribution is given as  $\hat{\Sigma} = \hat{R} \hat{S} \hat{S}^T \hat{R}^T$ .

The parametrization enables control over the Gaussians' position, scale, and rotation by manipulating the underlying triangle mesh. Applying transformations to the triangle directly alters the corresponding Gaussian.

**MiraGe** In this work, we present an approach that leverages the concept of flat Gaussian distributions in 3D space to model a single 2D image as input. Our methodology is grounded in human visual perception. This perspective allows us to reframe the problem: instead of merely processing a pixel matrix, we interpret the images as objects with a fixed spatial configuration in a 3D environment.

We put the 2D image on the  $XZ$  plane where the center is situated at axes origin  $(0, 0, 0)$  with the fixed distance from the camera origin. In practice, the distance from the plane is a hyper-parameter. In our approach, we model flat objects within 3D space, where the camera distance parameter effectively controls the perceived scale of the object. This relationship allows for intuitive adjustments of object size based on the desired visual effect. For instance, increasing the camera distance can naturally expand the apparent size of background elements like distant mountains (Fig. 4), making it easier to represent them as larger objects without additional modeling complexity. While this feature is beneficial, it is not strictly necessary for most applications.

We propose a method that situates the Gaussians within the  $XZ$  plane, ensuring that the entire image remains visible under perspective projection. To achieve this, the possible range of  $x$ -values and  $z$ -values is calculated using the camera field of view. We first calculate the deviation from 0 on the  $X$  axis using the similarity of triangles  $\text{dev}_z = \text{cam}_{\text{dist}} \cdot \tan(\frac{1}{2} \text{Fov}_{\text{vert}})$ , where  $\text{cam}_{\text{dist}}$  and  $\text{Fov}_{\text{vert}}$  are camera distance from the  $XZ$  plane and camera field of view respectively. The deviation in the  $X$  axis can be then

computed by multiplying this value by the camera aspect ratio.

Consequently, the initialization of Gaussians is consistently performed on the  $XZ$  plane; however, we have opted to permit their movement within the 3D space. Drawing inspiration from three distinct models, we introduce three conceptual approaches for manipulating the spatial positioning of Gaussians.

**Amorphous** The baseline approach how to control Gaussians is based on the classical GaMeS parametrization, initialized randomly on the  $XZ$  plane, with the mean parameter's  $y$  coordinate set to zero:

$$\mathcal{G} = \{(\mathcal{N}([m_1, 0, m_3], [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3], \text{diag}(\varepsilon, s_2, s_3)), \sigma_i, c_i)\}, \quad (2)$$

where  $\mathbf{m} = [m_1, 0, m_3]$ ,  $S = \text{diag}(s_1, s_2, s_3)$ , with  $s_1 = \varepsilon$ , and  $R$  is the rotation matrix defined as  $R = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$ , with  $\mathbf{r}_i \in \mathbb{R}^3$ .

It should be highlighted that we only initialized the Gaussian component on the  $XZ$  plane. During training, Gaussians can move amorphously in 3D space. We use the classical loss function  $L_1$  combined with a D-SSIM term:

$$\mathcal{L} = (1 - \lambda) \mathcal{L}_1(I, GS(I)) + \lambda \mathcal{L}_{D-SSIM}(I, GS(I)),$$

where  $I$  is the input image and  $GS(I)$  is the constraint obtained by the Gaussian renderer. While this solution enables the modeling of images using a collection of triangles, often referred to as "triangle soup," it proves insufficient for high-quality representations. During editing, significant artifacts emerge (Fig. 6–Baseline).

**2D** Building on the promising outcomes of GaussianImage we anchored all Gaussians to the  $XZ$  plane, translating flat image geometry into a spatial framework. We set the mean of these components to zero in the second coordinate. Moreover, we use the projection of flat Gaussians on a 2D plane. Unfortunately, orthogonal projection can produce artifacts. Therefore, we use a rotation of Gaussian components to lay on the  $XZ$  plane. Since we use flat Gaussians to extract such rotation, we can use a rotation matrix between two vectors to align the vector in 3D (Markley, 1993). We use the notation  $\text{Rot}(a, b)$  for the rotation matrix.

MiraGe on 2D plane is defined by set of 3D parameterized Gaussian components:

$$\mathcal{G} = \{(\mathcal{N}(\mathbf{m}, R_i \text{Rot}(\mathbf{r}_3, \mathbf{e}_2), S, \sigma_i, c_i)\},$$

where  $S = \text{diag}(s_1, s_2, s_3)$ , with  $s_1 = \varepsilon$ ,  $\mathbf{e}_2 = [0, 1, 0]$ ,  $\mathbf{m} = [m_1, 0, m_3]$ , and  $R_i$  is the rotation matrix defined as  $R_i = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$ , with  $\mathbf{r}_i \in \mathbb{R}^3$ .

**Graphite** Unfortunately, 2D-MiraGe produces artifacts when we use modification in 3D space (see the third row in

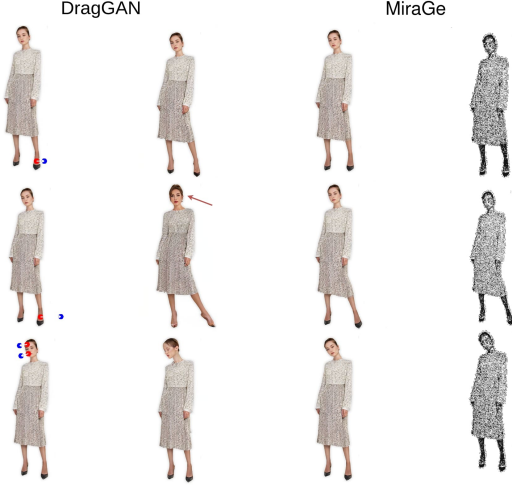


Figure 10. We compare the animation capabilities of MiraGe a DragGAN model, highlighting the advantages of our Gaussian-based image representation. This approach enables highly realistic edits by not relying on generative techniques. Our method offers greater control during animation. For example, adjusting the position of a leg does not inadvertently alter facial features.

Fig. 6). Such an effect is caused by the Gaussians, which appear randomly according to the camera position. To solve such a problem and obtain the possibility of 3D modifications, MiraGe allows the Gaussians to leave the  $XZ$  plane:

$$\mathcal{G} = \{(\mathcal{N}(\mathbf{m} + \gamma \mathbf{e}_2, R_i \text{Rot}(\mathbf{r}_3, \mathbf{e}_2)), S, \sigma_i, c_i)\},$$

where  $\gamma$  is trainable parameter of translation scale along the vector  $\mathbf{e}_2 = [0, 1, 0]$ ,  $\mathbf{m} = [m_1, 0, m_3]$ ,  $S = \text{diag}(s_1, s_2, s_3)$ , with  $s_1 = \varepsilon$ , and  $R_i$  is the rotation matrix defined as:  $R_i = [\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3]$ , with  $\mathbf{r}_i \in \mathbb{R}^3$ . Such a model allows for the order of Gaussians according to camera positions.

By leveraging parameterized Gaussians, we achieved precise manipulation of 2D images directly within their native 2D space, enabling targeted edits of segmented regions and transformations of complete scenes in an easier way. While this approach demonstrated substantial promise, we observed significant artifacts when extending manipulations into the 3D domain, particularly along the  $Y$ -axis, see first and last row in Fig 6.

**Mirror camera** We employ a novel approach utilizing two opposing cameras positioned along the  $Y$  axis, symmetrically aligned around the origin and directed towards one another. The first camera is tasked with reconstructing the original image, while the second models the mirror reflection. We introduced the mirror camera to ensure that Gaussians remain confined within a specific spatial region between the cameras, enhancing control and precision.

The reflection can be effectively represented by horizontally

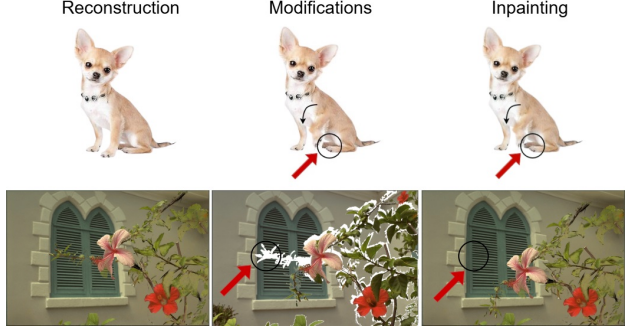


Figure 11. MiraGe model allows modifications in 3D space, but the model is limited by 2D images, which was used in training. When we move some elements from the foreground, we cannot see the background since the model only reconstructs objects. Next, we can use image Inpainting to fill the missing parts, allowing for more realistic modifications.

flipping the image, denoted as  $\mathcal{M}(I)$ . This mirror-camera setup enhances the fidelity of the generated reflections, providing a robust solution for accurately capturing visual elements. We consider the additional camera as a means of augmenting the dataset to improve the accuracy of the representation. The MiraGe is initialized according to equation Eqn. 2 and utilizes a cost function:  $\mathcal{L}(I) + L(\mathcal{M}(I))$ . We simultaneously model both the image and its mirrored reflection, as shown in the second row in Fig. 6. We provide numerical comparisons in the ablation study in the Appendix.

After thorough experimentation, we find that our model, Amorphous-MiraGe, utilizing a mirror camera, achieves state-of-the-art reconstruction results. This model demonstrates significant advantages over alternative methods in terms of both performance and outcome quality.

**Editability** The ability to manipulate Gaussians based on their spatial positioning empowers MiraGe to effectively edit 2D images. When utilizing a mirror camera, the quality of the resulting images is sufficiently high, enabling the parameterization and animation of Gaussians to significantly reduce artifacts. Our findings demonstrate that our model facilitates the animation of both segmented objects and entire scenes. Users can create manual animation or leverage automated processes using physics engines like Taichi\_elements or Blender (Fig. 3,7). To incorporate MiraGe with the 2D physics engine, we use 2D-MiraGe (Fig. 7). In Fig. 8, we demonstrate that our method can also be applied to edit more complex scenes, such as changing human expression.

We argue that the Graphite-inspired model allows the creation of attractive compositions made of multiple images that effectively present the positive attributes of the layered structure, like Graphite, through the strategic positioning of Gaussians.



Table 1. Quantitative comparison with various baselines in PSNR and MS-SSIM. MiraGe gives state-of-the-art results. Model- $x$  denotes that the model was initialized with  $x$  Gaussians. The metrics for the baselines were taken from (Zhang et al., 2024); however, we re-ran the experiments for GaussianImage (GI) -70K (GI-70K) and our model using a V100 GPU to ensure a more reliable comparison. Those experiments are denoted in the following table by asterisk next to the method name.

	Kodak dataset			DIV2K dataset		
	PSNR $\uparrow$	MS-SSIM $\uparrow$	Train Time(s) $\downarrow$	PSNR $\uparrow$	MS-SSIM $\uparrow$	Train Time(s) $\downarrow$
WIRE	41.47	0.9939	14339	35.64	0.9511	25684
SIREN	40.83	0.9960	6582	39.08	0.9958	15125
I-NGP	43.88	0.9976	491	37.06	0.9950	676
NeuRBF	43.78	0.9964	992	38.60	0.9913	1715
3DGS	43.69	0.9991	340	39.36	0.9979	481
GaussianImage-70k	44.08	0.9985	107	39.53	0.9975	121
GaussianImage-70k*	44.12	0.9985	116	39.53	0.9975	112
GaussianImage-100k*	38.93	0.9948	126	41.48	0.9981	120
MiraGe-70k; 5k iter (our)	49.07	0.9993	57	44.37	0.9989	75
MiraGe-100k; 5k iter (our)	51.04	0.9995	59	46.23	0.9992	79
MiraGe-70k; 30k iter (our)	57.41	0.9998	547	53.22	0.9996	789
MiraGe-100k; 30k iter (our)	59.52	0.9999	560	54.54	0.9998	946

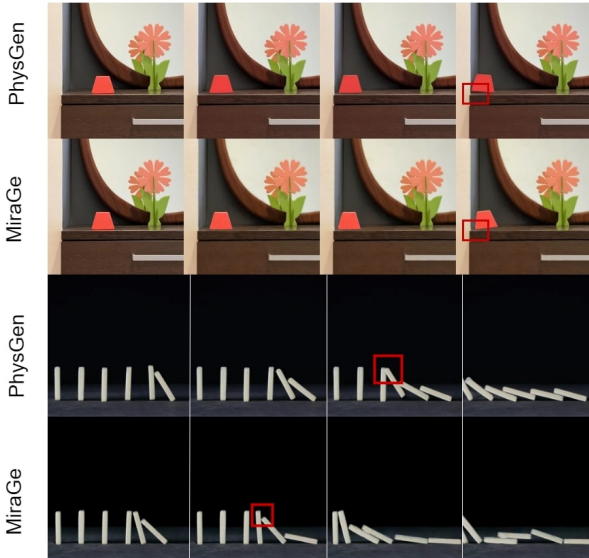


Figure 12. Comparison between PhysGen (Liu et al., 2025) and MiraGe. In the animation of a red block, reasonable doubts related to the correctness of the simulated physics of PhysGen arise when a reader follows the behavior of a red element upon hitting the wall; contrary to everyday experience, the front part (instead of the back part) of the figure bounces off the tabletop. In another animation, domino pieces start overlapping throughout the simulation. Our renders have properly solved the mentioned issues.

## 4. Experiments

The experimental section is split into two main parts. First, we demonstrate that our approach achieves high-quality 2D reconstruction by comparing it with existing models. Second, we highlight the versatility of MiraGe in image editing full scenes (Fig. 8, 13) and selected objects (Fig. 1), presenting examples of user-driven modifications and demonstrations involving physical simulations (Fig. 3, 7).

**Reconstruction quality** Our image reconstruction assessment utilizes two widely recognized datasets. Specifically, we employ the Kodak dataset<sup>4</sup>, which includes 24 images at a resolution of  $768 \times 512$ , alongside the DIV2K validation set (Agustsson & Timofte, 2017), which involves  $2 \times$  bicubic downscaling and comprises 100 images with sizes ranging from  $408 \times 1020$  to  $1020 \times 1020$ . The dataset was selected to facilitate direct comparison with the work of GaussianImage. As a baselines we use competitive INR methods GaussianImage (Zhang et al., 2024), SIREN (Sitzmann et al., 2020b), WIRE (Saragadam et al., 2023b), I-NGP (Müller et al., 2022), and NeuRBF (Chen et al., 2023).

In Tab. 1, we demonstrate the performance outcomes of different methods on the Kodak and DIV2K datasets. We see that our proposition outperforms the previous solutions on both datasets. The quality measured by both metrics shows significant improvement compared to all the previous approaches. Fig. 9 illustrates a general trend observed during training in the contest of image reconstruction. The selection of hyperparameters, including the number of iterations, was inspired by the principles of 3DGS. We provide ablation studies and extensive numerical analyses in the appendix for further insights.

Tab. 1 provides a comparison of training times across multiple baseline methods. For our approach, we report the full training durations corresponding to 30000 and 5000 iteration steps. Our method achieved better PSNR with only 5000 iterations, while requiring significantly less training time overall. Using the butterfly image from the DIV2K dataset as a case study, we showed that our method (Our-100K) surpasses GI in PSNR within just 30 seconds of training (Fig. 9).

<sup>4</sup><https://r0k.us/graphics/kodak/>



Figure 13. Although the scene is not fully modeled in 3D, representing a 2D image using 3D Gaussians enables novel viewpoints through camera movement. This creates a 2.5D effect, commonly used in games to simulate a background.

Fig. 9 includes the number of initial Gaussians, indicating how densely the space has been filled. We see a clear upward trend in performance as the density of the Gaussian initialization increases.

**Manual modification** MiraGe allows for manual manipulation of 2D images. By leveraging GaMeS parameterization, each Gaussian component is represented as a triangle. Vertices can be independently adjusted and moved within 3D space, enabling flexible image modification (Fig. 2).

We demonstrate examples of modifications using datasets such as DIV2K, Kodak, and Animals<sup>5</sup>. Additionally, we generated our own 2D images using DALL-E 3 to illustrate the benefits of our method. We can obtain modifications of small details like changing fingers’ position (Fig. 1), human facial expressions (Fig. 18), or dog poses (Fig. 11). As MiraGe can be trained in a 3D context, we can implement modifications in the third dimension to create the illusion of 3D transformation (Fig. 3, 6).

It is crucial to note that when we displace elements from the foreground, the background remains unseen because the model only reconstructs the objects. This is demonstrated in Fig. 11, where artifacts are apparent on the hind paw of the animal depicted. To reduce such a problem, we can use Inpainting (Perche-Mahlow et al., 2024) on the image background.

We conducted a comparative analysis of our editing approach against the DragGAN model (Pan et al., 2023). Here, we focus on the ability to perform localized edits, such as closing the mouth, while preserving other features, such as dimples (Fig. 8). The visual results, presented in Fig. 10, highlight key distinctions between the two models. As DragGAN is a generative model, modifications often result in unintended global transformations, for instance, attempting to adjust a leg’s position may inadvertently modify facial features. In contrast, our method demonstrates the ability to

move elements like the leg with realistic results and without compromising other aspects of the image.

**Physics application in MiraGe** Using 2D-MiraGe we can express Gaussian components with a 2D point cloud. Therefore, we can use an MPM (Hu et al., 2018) based physics engine implemented, for example, in Taichi\_elements. This high-performance physics engine supports multiple materials. We use inspiration from GASP (Borycki et al., 2024) and train simulation on 2D points, then use physical deformation on triangle soup. In Fig. 7, we present simulation results obtained using Taichi\_elements. As we can see, we can add physical properties to 2D objects. On the other hand, using Amorphous-MiraGe or Graphite-MiraGe, we can use Blender and modify directly parameterized flat 3D Gaussian (Fig. 3). Moreover, we compare MiraGe with PhysGen (Liu et al., 2025) (Fig. 12). Our method demonstrates better accuracy, ensuring that objects do not overlap. A key advantage is that users can directly and intuitively influence modifications.

**2.5D effect** MiraGe presents a concept that combines 2D and 3D representations to achieve the 2.5D effect commonly used in video games and VR (Feyer et al., 2024). Although we do not reconstruct full 3D geometry, our method leverages the third dimension for spatial manipulation and animation of 2D images. Fig. 13 illustrates this using a lantern image from the Kodak dataset, where camera movement reveals novel viewpoints. This 2.5D representation is especially useful for static backgrounds such as mountains in VR, where full 3D modeling is unnecessary. In such cases, our approach offers an efficient alternative to generative 3D models.

## 5. Conclusion

In this paper, we introduce MiraGe that uses flat 3D Gaussian components to model 2D images. MiraGe gives state-of-the-art reconstruction quality and simultaneously allows image manipulation. Furthermore, we can modify photos on a plane (Fig. 7) and in 3D space (Fig. 3). In consequence, we obtain the illusion of 3D-based modifications. Furthermore, we can combine our solution with a physics engine to obtain realistic motion in the image. Conducted experiments show that MiraGe is applicable in many different scenarios and produces high-quality simulations.

**Limitation** It is crucial to note that the model is not generative, so improper adjustment of Gaussian positions can cause gaps in the image (e.g. missing dog’s paw). This can be alleviated by using image Inpainting (Fig. 11). Although the model can produce realistic changes, a significant modification may introduce a visual artifact. Moreover, our model requires encoding more parameters than GaussianImage to achieve high-quality image reconstruction for animation.

<sup>5</sup><https://www.kaggle.com/datasets/alessiocrado99/animals10>

## Acknowledgements

The project “Effective rendering of 3D objects using Gaussian Splatting in an Augmented Reality environment” (FENG.02.02-IP.05-0114/23) is carried out within the First Team programme of the Foundation for Polish Science co-financed by the European Union under the European Funds for Smart Economy 2021-2027 (FENG). The work of P. Spurek was supported by the National Centre of Science, Poland Grant No. 2021/43/B/ST6/01456. The research of T. Szczepanik was funded by the program Excellence Initiative - Research University at the Jagiellonian University in Kraków.

## Impact statement

Editing images in pixel representation is a well-established technique with many existing solutions, and 2D image editing tools (e.g. Photoshop, GIMP). our introduces a representation that integrates 2D images with 3D to produce a 2.5D effect, a technique widely adopted in video games and VR application. The approach streamlines image reconstruction and manipulation without complex 3D generative models. It democratizes 3D editing, benefiting digital art, AR/VR and scientific visualization. By making 3D effects accessible and computationally efficient.

## References

- Agustsson, E. and Timofte, R. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 126–135, 2017.
- Bi, Z., Zeng, Y., Zeng, C., Pei, F., Feng, X., Zhou, K., and Wu, H. Gs<sup>3</sup>: Efficient relighting with triple gaussian splatting. In *SIGGRAPH Asia 2024 Conference Papers*, 2024.
- Björck, A. Numerics of Gram-Schmidt orthogonalization. *Linear Algebra and Its Applications*, 197:297–316, 1994.
- Borycki, P., Smolak, W., Waczyńska, J., Mazur, M., Tadeja, S., and Spurek, P. Gasp: Gaussian splatting for physics-based simulations. *arXiv preprint arXiv:2409.05819*, 2024.
- Chen, Z., Li, Z., Song, L., Chen, L., Yu, J., Yuan, J., and Xu, Y. Neurf: A neural fields representation with adaptive radial basis functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4182–4194, 2023.
- Davoodi, P., Ezoji, M., and Sadeghnejad, N. Classification of natural images inspired by the human visual system. *Neurocomputing*, 518:60–69, 2023.
- Feyer, S. P., Pinaud, B., Kobourov, S., Brich, N., Krone, M., Kerren, A., Behrisch, M., Schreiber, F., and Klein, K. 2D, 2.5D, or 3D? An Exploratory Study on Multi-layer Network Visualisations in Virtual Reality. *IEEE Transactions on Visualization & Computer Graphics*, 30(01):469–479, January 2024. ISSN 1941-0506. doi: 10.1109/TVCG.2023.3327402.
- Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., and Kanazawa, A. Plenoxels: Radiance fields without neural networks. In *The IEEE/CVF Computer Vision and Pattern Recognition Conference*, pp. 5501–5510, 2022.
- Gao, J., Gu, C., Lin, Y., Li, Z., Zhu, H., Cao, X., Zhang, L., and Yao, Y. Relightable 3d gaussians: Realistic point cloud relighting with brdf decomposition and ray tracing, 2024.
- Guédon, A. and Lepetit, V. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5354–5363, 2024.
- Hu, Y., Fang, Y., Ge, Z., Qu, Z., Zhu, Y., Pradhana, A., and Jiang, C. A moving least squares material point method with displacement discontinuity and two-way rigid body coupling. *ACM Transactions on Graphics (TOG)*, 37(4): 1–14, 2018.
- Huang, B., Yu, Z., Chen, A., Geiger, A., and Gao, S. 2d gaussian splatting for geometrically accurate radiance fields. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024.
- Jacobson, A., Baran, I., Popović, J., and Sorkine, O. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.*, 30(4), July 2011. ISSN 0730-0301. doi: 10.1145/2010324.1964973. URL <https://doi.org/10.1145/2010324.1964973>.
- Jiang, Y., Yu, C., Xie, T., Li, X., Feng, Y., Wang, H., Li, M., Lau, H., Gao, F., Yang, Y., and Jiang, C. Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. In *ACM SIGGRAPH 2024 Conference Papers*, SIGGRAPH ’24, New York, NY, USA, 2024. Association for Computing Machinery. ISBN 9798400705250. doi: 10.1145/3641519.3657448. URL <https://doi.org/10.1145/3641519.3657448>.
- Kerbl, B., Kopanas, G., Leimkühler, T., and Drettakis, G. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.



- Klocek, S., Maziarka, L., Wolczyk, M., Tabor, J., Nowak, J., and Smieja, M. Hypernetwork functional image representation. In *International Conference on Artificial Neural Networks*, pp. 496–510. Springer, 2019.
- Liu, K., Zhan, F., Xu, M., Theobalt, C., Shao, L., and Lu, S. Stylegaussian: Instant 3d style transfer with gaussian splatting. *arXiv preprint arXiv:2403.07807*, 2024a.
- Liu, S., Ren, Z., Gupta, S., and Wang, S. Physgen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision*, pp. 360–378. Springer, 2025.
- Liu, Z., Zhu, H., Zhang, Q., Fu, J., Deng, W., Ma, Z., Guo, Y., and Cao, X. Finer: Flexible spectral-bias tuning in implicit neural representation by variable-periodic activation functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2713–2722, 2024b.
- Lu, Y. Artificial intelligence: a survey on evolution, models, applications and future trends. *Journal of Management Analytics*, 6(1):1–29, 2019. doi: 10.1080/23270012.2019.1570365.
- Markley, F. L. Attitude determination using vector observations: A fast optimal matrix algorithm. In *Flight Mechanics (Estimation Theory Symposium, 1992, 1993)*.
- Moenne-Loccoz, N., Mirzaei, A., Perel, O., de Lutio, R., Esturo, J. M., State, G., Fidler, S., Sharp, N., and Gojcic, Z. 3d gaussian ray tracing: Fast tracing of particle scenes. *ACM Transactions on Graphics and SIGGRAPH Asia*, 2024.
- Müller, T., Evans, A., Schied, C., and Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4): 1–15, 2022.
- Pan, X., Tewari, A., Leimkühler, T., Liu, L., Meka, A., and Theobalt, C. Drag your gan: Interactive point-based manipulation on the generative image manifold. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings, SIGGRAPH ’23*, pp. 1–11. ACM, July 2023. doi: 10.1145/3588432.3591500.
- Perche-Mahlow, F. R., Felipe-Zanella, A., Cruz-Castañeda, W. A., and Amadeus, M. An inpainting-infused pipeline for attire and background replacement. *arXiv preprint arXiv:2402.03501*, 2024.
- Saragadam, V., LeJeune, D., Tan, J., Balakrishnan, G., Veeraraghavan, A., and Baraniuk, R. G. WIRE: Wavelet Implicit Neural Representations, 2023a. URL [https://openaccess.thecvf.com/content/CVPR2023/html/Saragadam\\_WIRE\\_Wavelet\\_Implicit\\_Neural\\_Representations\\_CVPR\\_2023\\_paper.html](https://openaccess.thecvf.com/content/CVPR2023/html/Saragadam_WIRE_Wavelet_Implicit_Neural_Representations_CVPR_2023_paper.html). [Online; accessed 3. Mar. 2024].
- Saragadam, V., LeJeune, D., Tan, J., Balakrishnan, G., Veeraraghavan, A., and Baraniuk, R. G. Wire: Wavelet implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18507–18516, 2023b.
- Shi, Y., Xue, C., Pan, J., Zhang, W., Tan, V. Y., and Bai, S. Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. *arXiv preprint arXiv:2306.14435*, 2023.
- Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7462–7473. Curran Associates, Inc., 2020a. URL [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf).
- Sitzmann, V., Martel, J., Bergman, A., Lindell, D., and Wetzstein, G. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020b.
- Tadeja, S. K., Solari Bozzi, L. O., Samson, K. D. G., Pattinson, S. W., and Bohné, T. Exploring the repair process of a 3d printer using augmented reality-based guidance. *Computers & Graphics*, 2023. ISSN 0097-8493. doi: 10.1016/j.cag.2023.10.017.
- Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., and Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 7537–7547. Curran Associates, Inc., 2020. URL [https://proceedings.neurips.cc/paper\\_files/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf).
- Waczyńska, J., Borycki, P., Kaleta, J., Tadeja, S. a., and Spurek, P. a. D-miso: Editing dynamic 3d scenes using multi-gaussians soup. In Globerson, A., Mackey, L., Belgrave, D., Fan, A., Paquet, U., Tomczak, J., and Zhang, C. (eds.), *Advances in Neural Information Processing Systems*, volume 37, pp. 107865–107889. Curran Associates, Inc., 2024.

- Waczyńska, J., Borycki, P., Tadeja, S., Tabor, J., and Spurek, P. Games: Mesh-based adapting and modification of gaussian splatting. *arXiv preprint arXiv:2402.01459*, 2024.
- Wang, Y., Jacobson, A., Barbivc, J., and Kavan, L. Linear subspace design for real-time shape deformation. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi: 10.1145/2766952. URL <https://doi.org/10.1145/2766952>.
- Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., and Jiang, C. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. *The IEEE / CVF Computer Vision and Pattern Recognition Conference*, 2024.
- Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V., and Sridhar, S. Neural fields in visual computing and beyond. In *Computer Graphics Forum*, volume 41, 2, pp. 641–676. Wiley Online Library, 2022.
- Yin, Z., Wang, Y., Papatheodorou, T., and Hui, P. Text2vrscene: Exploring the framework of automated text-driven generation system for vr experience. In *2024 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*, pp. 701–711, 2024. doi: 10.1109/VR58804.2024.00090.
- Zhang, X., Ge, X., Xu, T., He, D., Wang, Y., Qin, H., Lu, G., Geng, J., and Zhang, J. Gaussianimage: 1000 fps image representation and compression by 2d gaussian splatting. In *European Conference on Computer Vision*, 2024.

## A. Appendix

Here, we provide a future directions and a comprehensive overview of the implementation details. Furthermore, we present supplementary experimental results, such as extended performance evaluations and ablation studies focusing on camera settings.

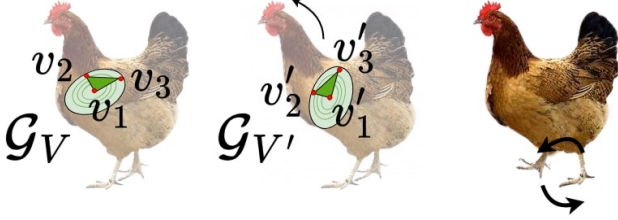


Figure 14. MiraGe use GaMeS (Waczyńska et al., 2024) representations of flat Gaussian by triangle soup. Therefore, we can use real-life modification by moving points.

### A.1. Future direction

Since our model is built on a flexible 3D Gaussian Splatting framework, it can partially adapt techniques from other models. We identify two promising directions for extending the method. For instance, StyleGaussian (Liu et al., 2024a) allows for color adjustments during style transfer. We believe that in practice, MiraGe can incorporate various style transfer methods for Gaussians. An important direction for future work is to investigate the adaptability of 3D Gaussians across diverse visual styles. Our model does not fully account for physically accurate lighting, leading to inconsistencies in light positioning. We believe that recent methods that incorporate real-world lighting (Gao et al., 2024; Bi et al., 2024) are transferable. Unlike BRDF-based models, 3D Gaussian Ray Tracing (Moenne-Loccoz et al., 2024) enables secondary lighting effects via ray tracing. Integrating such techniques into MiraGe is feasible and promising for future development.

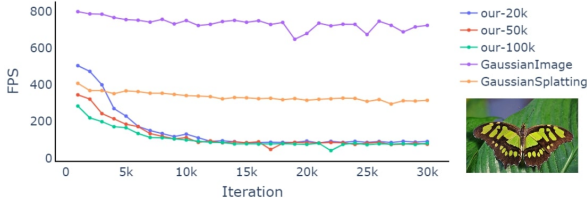


Figure 15. Comparison of FPS obtained on a butterfly image from DIV2K dataset by MiraGe in comparison with GaussianImage (Zhang et al., 2024) and Gaussian Splatting (Kerbl et al., 2023). The experiment was performed on the RTX 4070 GPU.

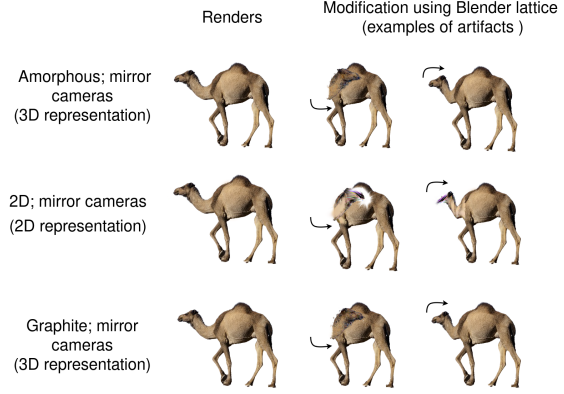


Figure 16. Example of artifacts generated during animation, typically due to imperfect rendering. The model was trained on a white background, leaving residual white Gaussians along the border of the camel’s muzzle, leading to artifacts. In this instance, the Graphine-MiraGe performed best in handling the head-turning movement

### A.2. Implementation details

The source code for this project is available under . Our code was developed based on the GaMeS framework (Fig. 14) and is distributed under the GS Vanilla license. Computational experiments in the main paper were conducted using NVIDIA GeForce RTX 4070 Laptop version and NVIDIA GeForce RTX 2080. Appendix time comparisons were reported using NVIDIA GeForce RTX 2080.

Building upon the GaMeS framework, we initialized the Gaussian distributions to lie perpendicular to the  $XZ$  plane. In our model, where all Gaussians are constrained to a 2D plane at rendering time, we consider only the rotation angle, denoted as  $\phi$ , as the primary rotation parameter. To facilitate the rendering of Gaussians positioned on the  $XZ$  plane,  $\phi$  serves as the primary learning parameter. The corresponding quaternions of rotation are computed as follows: for rotation about the x-axis  $q_x = [\cos(\frac{\phi}{2}), \sin(\frac{\phi}{2}), 0, 0]$ , and for the z-axis  $q_z = [\cos(\frac{\pi}{2}), 0, 0, \sin(\frac{\pi}{2})]$ . Since no rotation occurs about the y-axis, the quaternion remains  $q_y = [1, 0, 0, 0]$ . These quaternions are then combined through multiplication to form a new rotation matrix, ensuring precise alignment of the Gaussians on the  $XZ$  plane.

### A.3. Supplementary Numerical Findings from the Primary Paper

We conducted an extensive analysis of the MiraGe model due to its unique ability to control the behavior of Gaussians. Three distinct settings for Gaussian movement were explored:

- Amorphous the first allows Gaussians to move freely



Table 2. Training time and memory usage of MiraGe across varying image resolutions. For Training we used 100K initial Gaussians; 5k, 10k, 30k iterations on the V100 GPU. Experiment is provided on a *butterfly* from the DIV2K dataset.

Model	Our-5k iter			Our-10k iter			Our-30k iter			GaussianImage/default settings		
scale	PSNR	MB	Train time	PSNR	MB	Train time	PSNR	MB	Train time	PSNR	MB	Train time
1/none	30.76	3.56	209	36.17	12.2	586	47.04	35.65	2709	28.34	2.41	189
2	43.85	2.54	67	47.21	4.20	158	52.27	8.24	651	38.54	2.41	111
3	51.51	2.41	45	54.20	3.53	104	58.81	5.66	406	41.53	2.41	105
4	56.86	2.36	40	58.52	3.27	92	65.02	4.83	341	34.33	2.41	103

Table 3. Ablation study of the effect of adding the mirror camera as augmentation technique on training time and the output image quality measured in widely recognized metrics: PSNR, MS-SSIM, LPSIS. The experiment was performed with an initial 100k Gaussians.

Kodak dataset					
Gaussian control method	Camera Setting	PSNR $\uparrow$	MS-SSIM $\uparrow$	LPSIS $\downarrow$	Training Time(s) $\downarrow$
Amorphous	One camera	51.56	0.9996	0.0050	448.73
	Mirror cameras	59.52	0.9999	0.0005	639.66
Graphite	One camera	42.49	0.9948	0.2984	398.54
	Mirror cameras	46.90	0.9983	0.1238	739.66
2D	One camera	42.75	0.9950	0.2931	552.80
	Mirror cameras	48.82	0.9987	0.0071	942.78
DIV2K dataset					
Gaussian control method	Camera Setting	PSNR $\uparrow$	MS-SSIM $\uparrow$	LPSIS $\downarrow$	Training Time(s) $\downarrow$
Amorphous	One camera	46.00	0.9991	0.0162	690.98
	Mirror cameras	54.54	0.9998	0.0033	946.35
Graphite	One camera	40.02	0.9949	0.0312	582.50
	Mirror cameras	46.52	0.9986	0.0117	1082.41
2D	One camera	39.99	0.9949	0.0310	869.62
	Mirror cameras	46.32	0.9985	0.0124	1278.33

in 3D space,

- 2D: the second restricts their movement to align parallel to the  $XZ$  plane
- Graphite the third confines all Gaussians to the  $XZ$  plane, effectively creating a 3D representation.

A qualitative analysis was performed, considering the impact of the mirror camera (Tab. 3), as well as the effect of varying the number of initial Gaussians on the overall model behavior (Tab. 4). We also examined the impact of the camera using the Frames Per Second (FPS) metric and storage memory (Tab. 5). Given the ongoing development of various 3D Gaussian Splatting compression techniques, we employed the `.spz`<sup>6</sup> tool to effectively compress the data.

Due to our particular focus on animation, we analyzed FPS trends to benchmark real-time performance. Fig. 15 shows that while our model introduces a higher number of parameters, leading to a decrease in FPS compared to GaussianImage, it maintains the ability to render animations in

real-time.

Tab. 2 presents an analysis of our model’s scalability. We selected a butterfly image from the DIV2K dataset (Fig. 9), which provides official rescaling, to evaluate performance across different resolutions. Our model was trained on each rescaled version and compared with GaussianImage, a only Gaussian-based baseline. Even with only 5000 training iterations, our method learns faster and achieves higher PSNR. Across all scales, our approach consistently outperforms the baseline in terms of reconstruction quality.

Tab. 3 shows the mirror camera view as the augmentation technique significantly improves the representation’s fidelity of every proposed Gaussian method. This behavior can be detected with the help of any of the measured metrics, i.e., PSNR, MS-SSIM and LPSIS. The drawback of improving the image quality is that a longer training time is required. The ablation study presented in Tab. 4 similarly suggests that our model scales well with the number of Gaussians used during model initialization. The striking example here is an average 62.12 PSNR score achieved by the Amorphous method on the Kodak dataset. The price paid in time of

<sup>6</sup><https://github.com/nianticlabs/spz>

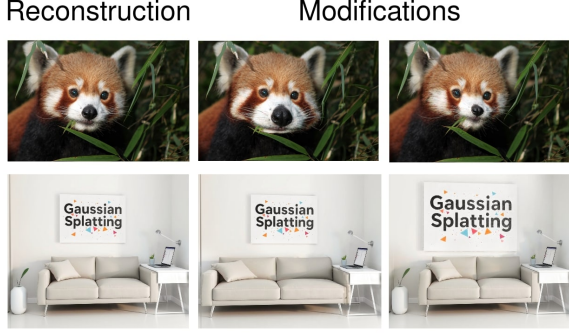


Figure 17. MiraGe enables modifying 2D images, such as adjusting the scene’s elements’ sizes.

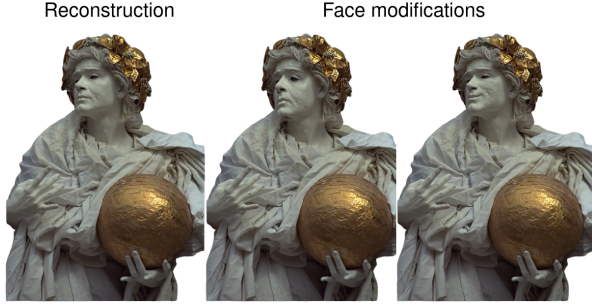


Figure 18. MiraGe allows us to produce realistic modifications of small details like changing human facial expressions.

training grows here slower, i.e., increasing the number of starting Gaussians by an order of magnitude results in more extended though comparable training period length.

#### A.4. Extension of examples modification and Artifacts

Animating a full scene can be non-trivial, but it is possible. Fig. 17 demonstrates how a painting can be enlarged to visualize the impact of its placement in a room, offering a clear view of the potential arrangement. It is also possible to animate small, localized areas of the image, as demonstrated in Fig. 18. For the facial animation, we utilized the Lattice modifier in Blender. A simple editing concept using 3D is shown in Fig 19. Fig. 20 illustrates a sculpture where the movement of the hand is achieved by adjusting the position of the shield behind the warrior. The image representation, based on parameterized Gaussians, facilitates precise editing of fine details within the 3D space. MiraGe enables manual image editing and incorporates a physics engine for image modifications (Fig. 21, 22, 20). It is crucial to remember that if certain Gaussians are shifted without considering their dependencies on others, the image will be disrupted. Therefore, the relationships between the Gaussians must be carefully modeled. We demonstrate this concept with the example of children playing with a blanket (Fig. 22). Despite the movement of the blanket (as seen in the supplementary video), the image remains uninterrupted and coherent.

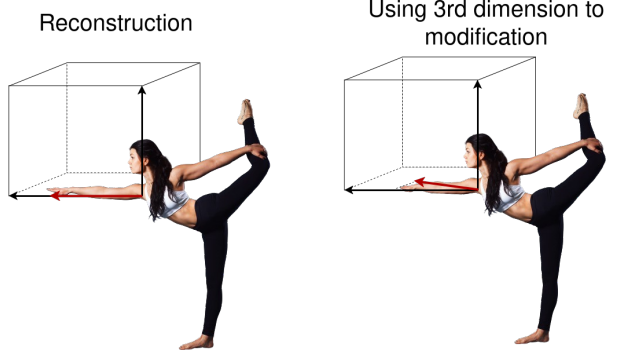


Figure 19. MiraGe simplifies intuitive editing of images, allowing transformations such as adjusting the tilt of a hand with minimal complexity. This is achieved by modifying the object along the third dimension.

Integrating the representation into Blender can introduce automatic adjustments that may result in visual artifacts (Fig. 16), particularly when training on images with a white background. These modifications can lead to unrealistic renderings that are challenging to detect through automated means and currently require subjective evaluation by a human observer.

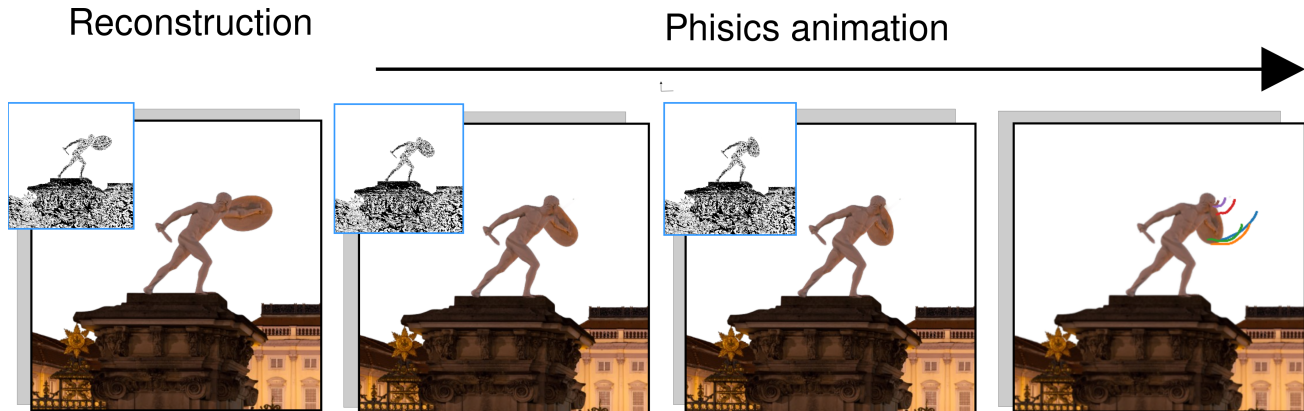


Figure 20. MiraGe can be integrated with Blender, by using flat 3D Gaussians in 3D space. The initial column presents the original image, the subsequent two columns display renders captured midway through the simulation, and the final column shows the outcome at the simulation’s conclusion. The colored lines in the last column trace the paths of 10 randomly chosen points from the simulation.

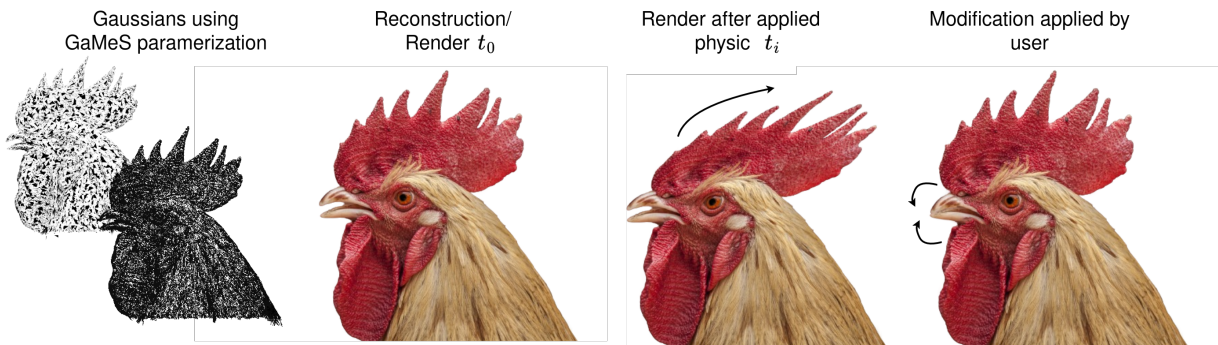


Figure 21. MiraGe allows for manual image edits and for using a physics engine for real-life-like image modifications. The left image illustrates a Gaussian representation achieved through a triangle mesh triangle soup, while the accompanying point-based depiction provides finer details, offering a more refined visual comparison.

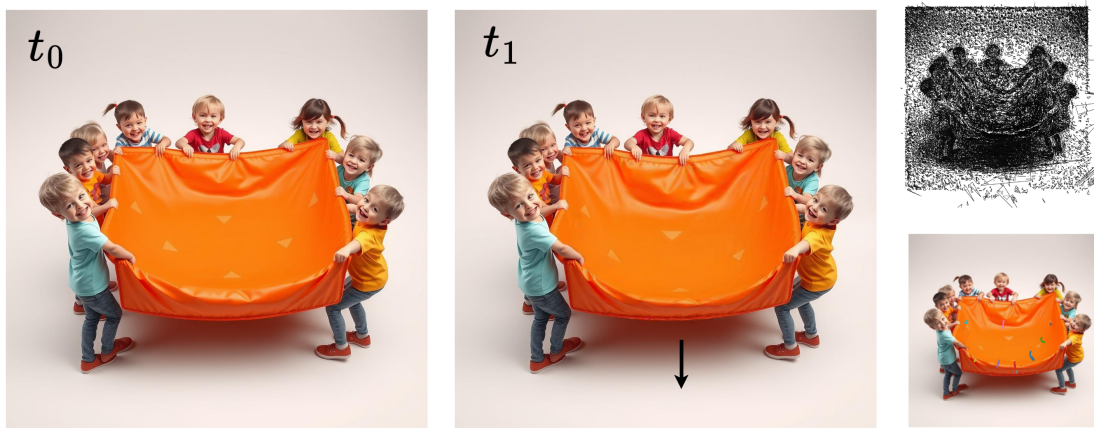


Figure 22. MiraGe allows for the modification of larger scenes. We can selectively alter specific areas and introduce smooth movements or material adjustments. In this example, the bottom of the blanket is shown in motion. This, along with other modifications, is available in the supplementary files as videos.



Table 4. Measuring the influence of the initial number of Gaussians on the image reconstruction quality. The experiment was performed using a mirror camera view for every table entry.

Kodak dataset					
Gaussian control method	Initial Gaussians	PSNR $\uparrow$	MS-SSIM $\uparrow$	LPSIS $\downarrow$	Training Time(s) $\downarrow$
Amorphous	10k	50.66	0.9987	0.3531	584.84
	50k	55.54	0.9997	0.0033	634.65
	100k	59.52	0.9999	0.0005	639.66
	150k	62.12	0.9999	0.0002	676.10
Graphite	10k	40.39	0.9940	0.0599	651.32
	50k	44.90	0.9973	0.2024	732.91
	100k	46.90	0.9983	0.1238	739.66
	150k	48.16	0.9987	0.0105	801.18
2D	10k	39.75	0.9886	0.0769	857.30
	50k	45.03	0.9955	0.2789	876.56
	100k	48.82	0.9987	0.0071	942.78
	150k	50.54	0.9992	0.0031	955.86
DIV2K dataset					
Gaussian control method	Initial Gaussians	PSNR $\uparrow$	MS-SSIM $\uparrow$	LPSIS $\downarrow$	Training Time(s) $\downarrow$
Amorphous	10k	49.53	0.9987	0.0322	852.19
	50k	52.23	0.9995	0.0112	902.80
	100k	54.54	0.9998	0.0033	946.35
	150k	56.40	0.9999	0.0014	975.44
Graphite	10k	40.75	0.9959	0.0457	983.41
	50k	44.67	0.9980	0.0216	1008.52
	100k	46.52	0.9986	0.0117	1082.41
	150k	47.61	0.9989	0.0083	1103.69
2D	10k	38.40	0.9920	0.0616	1166.09
	50k	42.86	0.9967	0.0275	1256.62
	100k	46.32	0.9985	0.0124	1278.33
	150k	48.46	0.9990	0.0065	1415.54

Table 5. Ablation study of the effect of adding the mirror camera as augmentation technique on Kodak dataset measured using Frames Per Second (FPS) and memory storage.

Kodak dataset				
Gaussian control method	Camera Setting	FPS	Memory (MB)	Compressed memory (MB)
GaussianImage-70k	-	-	-	2.41
GaussianImage-100k*	-	-	-	3.44
Amorphous	One camera	583.28	31.25	2.42
	Mirror cameras	620.10	117.25	7.80
Graphite	One camera	1157.75	30.71	2.68
	Mirror cameras	650.75	117.91	9.22
2D	One camera	1130.08	30.69	2.68
	Mirror cameras	418.39	173.64	12.82