GaussianDiffusion: Learning Image Generation Process in GaussianRepresentation Space

Simon Coessens*
CentraleSupélec
Gif-sur-Yvette, France
simon.coessens@proton.me

Arijit Samal*
CentraleSupélec
Gif-sur-Yvette, France
samalarijit.01@gmail.com

Akash Malhotra
Université Paris-Saclay
Orsay, France
akash.malhotra@universite-paris-saclay.fr

Nacéra Seghouani
CentraleSupélec
Gif-sur-Yvette, France
seghouani@lisn.fr

Abstract

Diffusion models have become a leading approach in generative image modeling, but many still operate in dense pixel space, a representation that is computationally intensive and lacks geometric structure. We propose GaussianDiffusion, a framework that performs the denoising process entirely in a latent space composed of 2D Gaussians. Each image is encoded as a set of 150 anisotropic Gaussian splats, parameterized by position, covariance, and color. To model their dynamics, we introduce GaussianTransformer, a permutation-equivariant transformer that serves as the denoising network. Evaluated on MNIST and Sprites datasets, our method achieves visual quality comparable to a pixel space U-Net baseline, while reducing the number of sampling steps from 1000 to 200 and the per-step cost from 11.4 GFLOPs to 4 GFLOPs, resulting in an overall 22× improvement in generation time on an A100 GPU. In contrast to latent diffusion models, our approach does not require an auxiliary autoencoder and preserves full editability of the latent. These findings suggest that structured geometric representations can offer efficient and interpretable alternatives to latent and pixel-based diffusion.

1. Introduction

Diffusion models have advanced generative vision forward, but many such models operate in *pixel grids* that ignore the rich geometric structure of images [3, 12]. This choice bloats computation, $H \times W$ values per step, and offers no built-in equivariance to translation or rotation. On a 512×512 image, a standard U-Net based DDPM costs $\approx$ 3000 GFLOPs and ≈ 1.4s per reverse step[1].

Recent "Gaussian Splatting" encodes a scene as a compact set of anisotropic Gaussians and renders it in $\mathcal{O}(N)$ time [4, 14]. We adopt this idea for *2D images*: an image becomes a collection of $N \ll H \times W$ splats with position, covariance, and color. The representation is inherently continuous, compresses aggressively, and is geometrically interpretable. Further, translation and rotation act directly on the parameters of the gaussians.

We run the diffusion process on *Gaussian Image Representation (GIR)*. A lightweight *GaussianTransformer* models their dynamics as the ϵ_θ network. Qualitatively, our generated samples match the visual fidelity of a pixel-space DDPM while requiring only $\sim$200 reverse steps versus the U-Net's $\sim$1000. Moreover, GaussianDiffusion stabilizes much earlier in training, by epoch 10 it already produces coherent sprite samples where the pixel model remains noisy (Figure 8). These improvements arise solely from the structured splat prior, no extra data, parameters, or auxiliary autoencoders are employed.

Because each timestep refines the gaussians, the reverse process produces a "materialization" effect of blobs into shapes. The effect shows the reverse diffusion process unfolding and enables direct, editable control over position, scale, and color.

Contributions.

1. We propose *Gaussian Image Representation* (GIR), a structured latent format in which images are represented as sets of 2D Gaussians. GIR introduces geometric inductive bias, improves training stability, and enables faster convergence in diffusion models.

*Equal contribution

[1] Based on our calculations for 32x32 Sprites dataset

2. We develop *GaussianTransformer*, a transformer-based denoising network tailored for GIR. Compared to conventional U-Net architectures operating in pixel space, it achieves significantly lower per-step computational cost while maintaining sample quality.

The rest of the paper proceeds as follows: Section 2 surveys related efforts on structural priors, diffusion, and Gaussian rendering; Section 3 details our model; Section 4 reports experiments; Section 5 concludes and outlines future work.

2. Related Work

Diffusion models and the missing structure. Denoising Diffusion Probabilistic Models (DDPM) [3] and their deterministic variant DDIM [13] introduced a powerful generative paradigm, typically implemented using U-Net architectures [11] in pixel space. While effective, these models operate over dense $H \times W$ grids, leading to high compute and limited structural inductive bias. Diffusion Transformers (DiT) [8] replace the U-Net with a global attention mechanism, but still inherit the structural vacuum of the pixel grid.

Latent diffusion and compression tricks. Latent Diffusion Models (LDM) [10] reduce the computational burden by performing diffusion in a VAE-learned latent space. This offers an approximate 8× speed-up but introduces a large (300M-parameter) autoencoder and encodes semantics into an opaque codebook, limiting interpretability and editability [6]. Other compression strategies, such as wavelet-based latents [2], exhibit similar limitations: the latent lacks explicit geometry and requires additional training overhead for the encoder-decoder.

Gaussian splatting as a structured latent. Gaussian Splatting encodes scenes using sets of anisotropic 3D Gaussians, enabling efficient rendering in $\mathcal{O}(N)$ time [4]. Recent work has extended this idea to 2D image representations [7, 14]. Gaussian Image Representation (GIR) inherits translation and rotation equivariance, offers strong compression ($N \ll H \times W$), and exposes interpretable parameters such as position, covariance, and colour. To our knowledge, no prior work combines 2D Gaussian splats with diffusion models; concurrent 3D efforts have focused on view synthesis [5].

Transformers for structured sets. Vision Transformers (ViT) [1] treat an image as a sequence of fixed patches; DiT tokens follow a similar design. In contrast, our *GaussianTransformer* treats each splat as a token. Since transformers are permutation-equivariant, they can directly operate on the unordered GIR set without requiring positional encodings, while self-attention captures pairwise relationships. This approach is conceptually aligned with Point Transformer [15], which applies attention to point clouds by leveraging geometric locality within sets.

Gap and our position. To summarise, existing diffusion models typically either operate directly in pixel space, which lacks geometric structure, or in learned latent spaces, which can obscure semantics and interpretability. Gaussian splats offer a structured and interpretable alternative, and transformers are a natural choice for processing such unordered sets. In this context, our work introduces a diffusion model that operates in the space of 2D Gaussians, combining these two ideas to reduce compute while preserving semantic clarity and generation quality.

3. GaussianDiffusion Method

GaussianDiffusion builds upon two key components: (i) a structured image representation called *Gaussian Image Representation (GIR)*, where each image is expressed as a set of analytic 2D Gaussians; and (ii) a diffusion model operating in this space, parameterised by a transformer architecture called *GaussianTransformer*. To isolate the contribution of GIR, we benchmark against a standard pixel-based DDPM using a U-Net with a comparable parameter count.

3.1. Gaussian Image Representation (GIR)

We represent each image as a fixed-size, unordered set:

$$G = \{g_i = (\mu_i, \Sigma_i, c_i)\}_{i=1}^{N}, \quad N = 150 \tag{1}$$

where $\mu_i \in \mathbb{R}^2$ denotes the 2D position, $\Sigma_i \in \mathbb{R}^{2\times 2}$ the covariance matrix, and $c_i \in \mathbb{R}^3$ the RGB colour of each Gaussian. Each element encodes 8 parameters, resulting in a total dimension of 1200 per image, which is significantly less than the 3072 dimensional 32×32×3 pixel space.

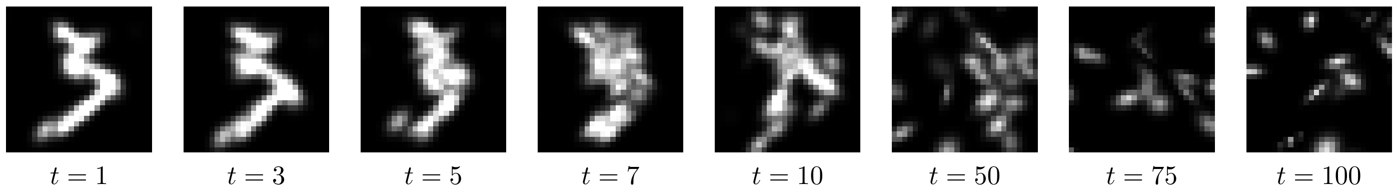
Gaussian Encoder and Decoder. To encode an image $I(x, y)$, we fit a mixture of Gaussian splats:

$$\hat{I}(x, y) = \sum_{i=1}^{N} w_i f_i(x, y \mid \mu_i, \Sigma_i) \tag{2}$$

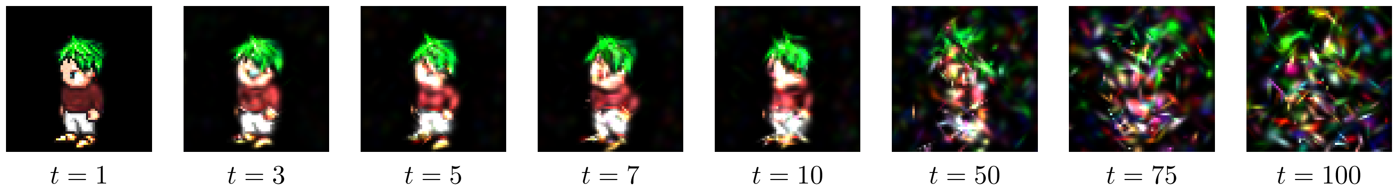
Each kernel f_i is defined as:

$$f_i(x, y) = \frac{1}{2\pi\sqrt{|\Sigma_i|}} \exp\left(-\frac{1}{2}\mathbf{v}_i^\top \Sigma_i^{-1} \mathbf{v}_i\right), \tag{3}$$

with $\mathbf{v}_i = \begin{bmatrix} x - \mu_{i,x} \\ y - \mu_{i,y} \end{bmatrix}$. Parameters $(\mu_i, \Sigma_i, c_i, w_i)$ are optimized using a combination of L1 and DSSIM losses. The

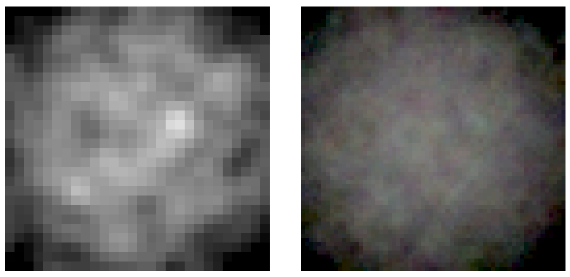


a) GIR Forward diffusion process for MNIST



b) GIR Forward diffusion process for Sprites

Figure 1. Forward diffusion processes for MNIST and Sprites at various timesteps.



(a) MNIST (b) Sprites

Figure 2. Comparison of terminal distributions for MNIST and Sprites datasets (Average of 100 images) in GIR.

decoder reconstructs the image via alpha blending of Gaussian layers.

The structured evolution of noise in this representation is shown in Figure 1, with distinct patterns visible across timesteps for both MNIST and Sprites. Terminal states across datasets are visualized in Figure 2, highlighting the concentrated and desaturated nature of fully noised GIRs.

3.2. Baseline: Pixel-Space U-Net DDPM

We implement a DDPM baseline using a conventional U-Net architecture [11], operating directly in 32×32 pixel space. The model comprises downsampling and upsampling paths with skip connections and sinusoidal time embeddings, totaling 38 million parameters. Figure 3 illustrates the U-Net structure. This setup provides a fair comparison for evaluating the benefits of structured representations in GaussianDiffusion.

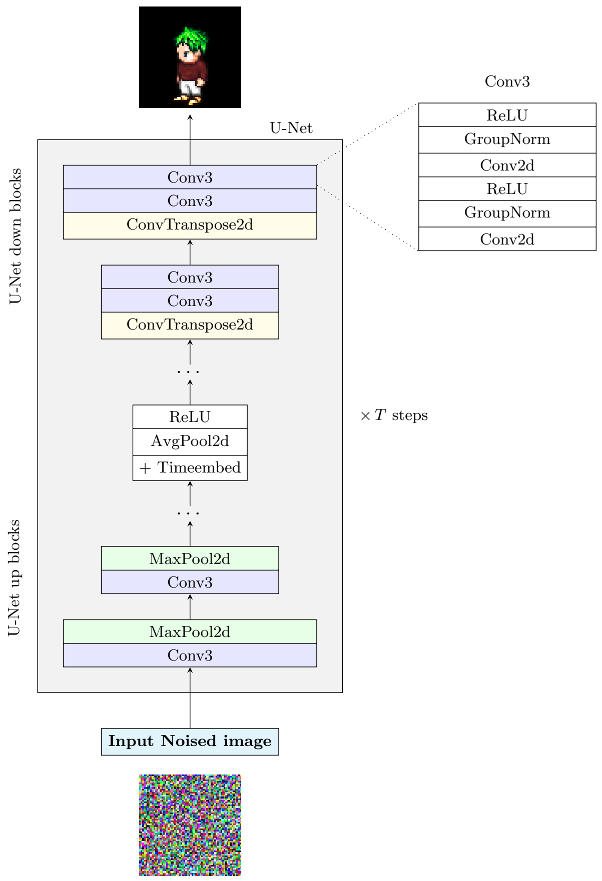


Figure 3. Baseline U-Net Architecture

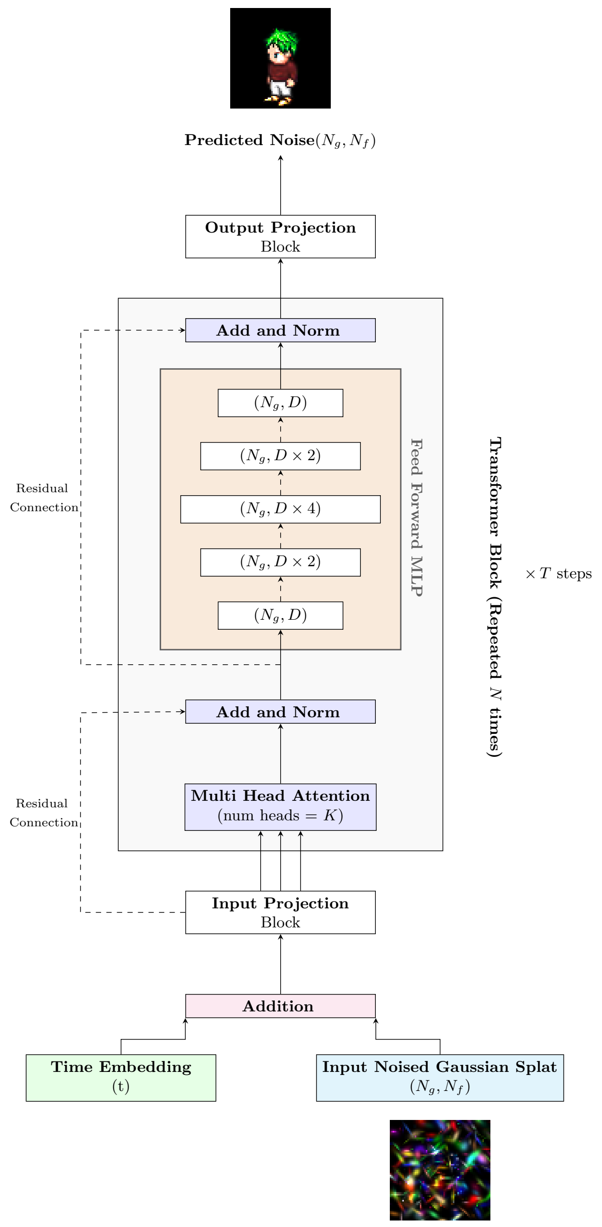


Figure 4. GaussianTransformer Architecture.

3.3. Diffusion in GIR

We follow the DDPM formulation [3], applying Gaussian noise at each forward step:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, I) \tag{4}$$

Here, $\mathbf{x}$ is the flattened vector of all GIR parameters. Because the noise is injected into positions, covariances, and colors, the terminal distribution converges to a smooth, grey blob centered in the frame. This low-entropy end state facilitates stable reverse diffusion, which we learn using an ℓ_2 noise-prediction loss. No additional stabilisation or custom scheduling was required.

3.4. GaussianTransformer

To model the reverse process in GIR space, we introduce *GaussianTransformer*, which treats each Gaussian splat as a token in an unordered set. Given input $G_t \in \mathbb{R}^{N\times F}$ (with $N = 150$, $F = 8$), the model predicts the denoised estimate $\hat{\epsilon}_\theta(G_t, t) = \{\hat{g}_i^{\text{noise}}\}_{i=1}^{N}$.

Architecture Overview. The model, illustrated in Figure 4, comprises:

- *FiLM-based Time Conditioning.* Timestep t is passed through a sinusoidal embedding, then modulates activations via Feature-wise Linear Modulation (FiLM) [9], ensuring adaptive depth-wise conditioning throughout the network.
- *Input Projection.* Each 8D Gaussian token is projected to 512D via a two-layer MLP with SiLU activation and LayerNorm.
- *Core Transformer Blocks.* The main body consists of 8 transformer layers, each using 8 head self-attention and a structured MLP stack (2×4×2×1). The architecture is fully permutation-equivariant, aligning naturally with the set nature of GIR.
- *Output Projection.* Each 512D token is mapped back to an 8D vector representing predicted noise for that splat.

The full model has approximately 40 million parameters, comparable to the U-Net baseline. During sampling, GaussianTransformer progressively refines the noisy GIR back into a coherent image.

4. Results and Evaluation

We report results on two canonical benchmarks: grayscale **MNIST** and the RGB **Sprites** character dataset. All Sprites images are down-sampled to 32×32 to match MNIST scale and to enable training under limited computational resources while maintaining a resolution-controlled comparison.

4.1. Datasets and GIR statistics

Table 1 summarizes the Gaussian Image Representation used during training. We encode *every* image with the same fixed N=150 splats and F=8 features, resulting in a compact $N \times F$ = 1200D latent regardless of dataset.

4.2. Experimental protocol

Both models are trained for the same wall-clock budget (24 GPU-hours on a single NVIDIA A100-40GB) using AdamW with identical learning-rate schedules. All timing numbers are measured in standard float32 precision on a

Dataset	Res.	Gauss./img	Compr. Ratio[†]	PSNR (dB)
MNIST	28×28	150	2.0	30.1
Sprites	32×32	150	2.7	36.4

Table 1. **GIR encoding statistics.** [†]Compression is defined as $\frac{HWC}{NF}$, i.e. pixel count divided by splat parameters.

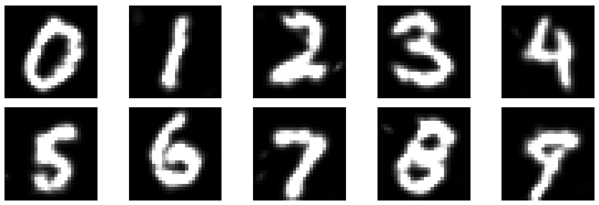


Figure 5. Generated MNIST samples using our GaussianTransformer model with 200 refinement steps. Note the clarity and diversity of the generated digits.

single A100 with batch size 32 and include only the forward pass per diffusion step.

Table 2 reports per-step latency and total generation time. GaussianDiffusion completes sampling in 200 steps at 1.2 ms/step (total $\approx$ 0.24 s), whereas the pixel U-Net requires 1000 steps at 5.4 ms/step (total $\approx$ 5.4 s), yielding a 22× reduction in end-to-end runtime.

4.3. Qualitative results

Figure 5 and Figure 7 show randomly sampled outputs after 200 reverse steps. The samples are diverse and match the training images in quality and resemblence.

4.4. Training Stability

Figure 8 compares 2×4 sampling grids from the pixel diffusion model (left) and GaussianDiffusion (right) at epochs 10, 20, and 50. The pixel model shows unstable, noisy samples in early epochs, whereas our method produces coherent outputs from epoch 10 onward, indicating faster and more stable convergence.

4.5. Sampling efficiency

The gains stem from three factors:

1. **Smaller latent.** GIR reduces the state from 3072 pixels to 1200 splat parameters.
2. **Stronger prior.** Gaussians, which have stronger geometric priors, guide the model, so it converges in five times fewer steps.
3. **GPU-friendly architecture.** Self-attention GEMMs saturate tensor cores; the U-Net's many 3×3 convolutions do not, widening the wall-clock gap beyond the raw FLOP ratio (see Table 2).

4.6. Gaussian materialisation

Operating in parameter space yields an interpretable *materialisation* effect: during reverse diffusion blurred blobs sharpen and coalesce into semantic parts as shown in Figure 6. This side-effect, while aesthetic, also allows frame-accurate editing of position, scale, and colour, illustrating the practical advantages of a structured latent space.

5. Conclusion

GaussianDiffusion demonstrates that replacing the pixel grid with a structured set of 2D Gaussians offers a viable alternative for generative modeling. By conducting the entire diffusion process in this splat space and predicting noise with a permutation-equivariant GaussianTransformer, we reduce generation steps from 1000 to 200 while cutting per-step cost from 11.4 GFLOPs (5.4 ms) to 4 GFLOPs (1.2 ms) on an A100 GPU. Training is also more stable: coherent digits and sprites emerge within ten epochs, in contrast to the noisier convergence of the pixel-space baseline.

While we do not report quantitative fidelity metrics such as FID or KID due to resource constraints, visual samples show comparable perceptual quality to the baseline. Our aim is not to compete on benchmarks but to offer a clearer and more interpretable pathway for structured diffusion.

Looking ahead, two extensions are particularly promising. First, integrating the 2D Gaussian encoder and decoder in an end-to-end training pipeline would allow splat parameters to co-evolve with the denoiser. Second, replacing the core transformer with specialized geometric attention modules, such as those from the Point Transformer family, may better capture the spatial interactions inherent in Gaussian sets. These directions further the SP4V goal of bridging structure and efficiency in scalable generative vision.

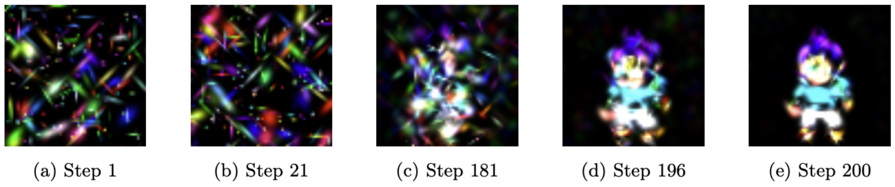


Figure 6. An example of reverse diffusion process (Gaussian Materialization).

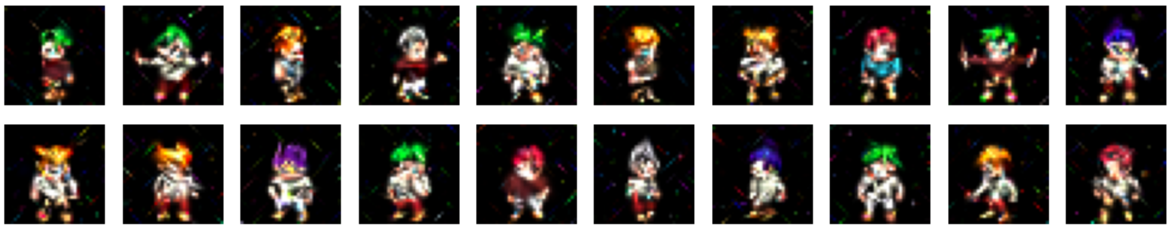
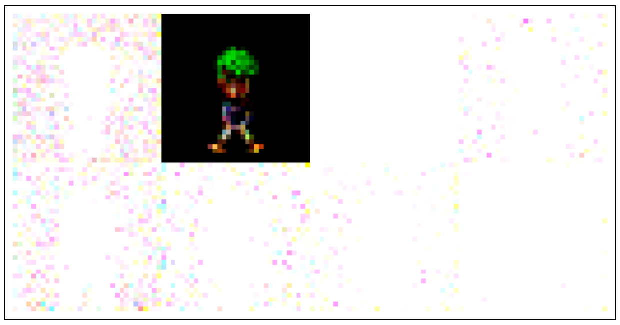


Figure 7. Generated Sprites samples using our GaussianTransformer model with 200 refinement steps. Note the clarity and diversity of the generated characters.

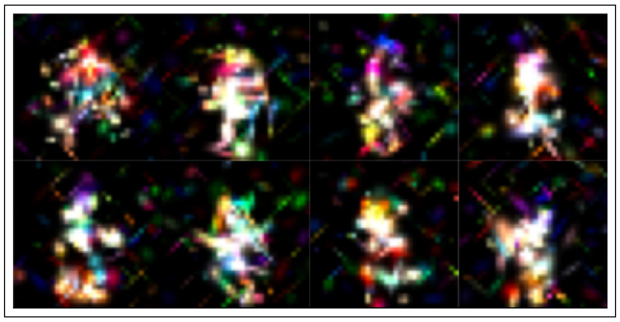
Model	**Params** (M)	**FLOPs** / step (G)	**Wall** / step (ms)‡	**Steps** (T = gen)	**Total** FLOPs (T)	**Total** time (T)
GaussianTransformer	≈40	4.0	1.2	200	0.80 T	0.24 s
U-Net (pixel)	≈38	11.4	5.4	1000	11.4 T	5.40 s

‡FP16/TF32 mixed precision, batch 32, single NVIDIA A100-40GB. Backward pass cost is ≈2× forward.

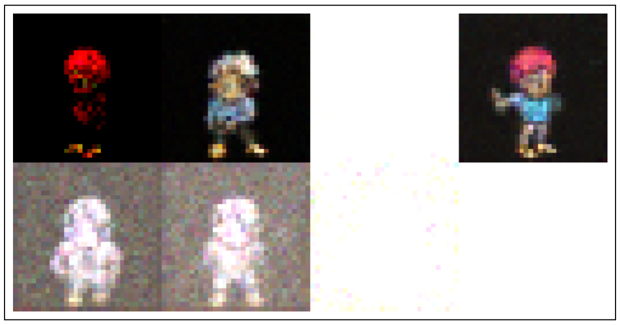
Table 2. Compute profile at 32 × 32 resolution. GaussianDiffusion uses **2.8×** fewer FLOPs per step and, thanks to transformer tensor-core utilisation, runs **4.5×** faster wall-clock; combined with the 5× lower step count, total generation time drops from 5.4s to 0.24s (a 22× speed-up).



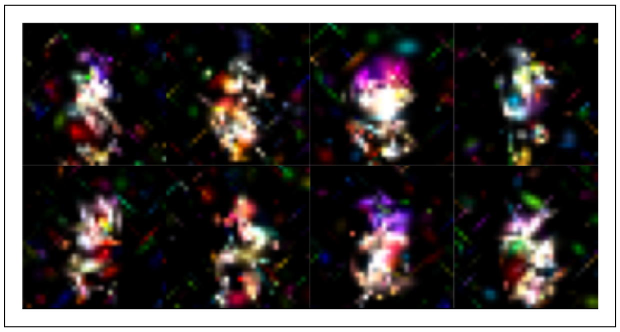
(a) Epoch 10



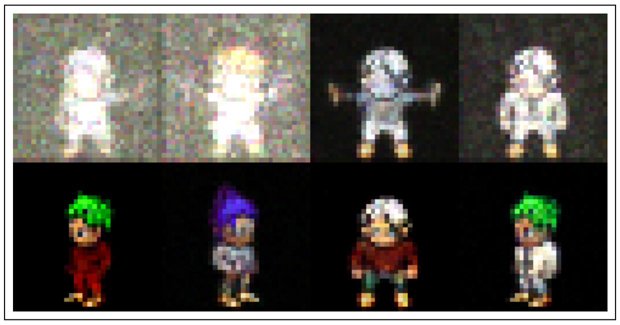
(b) Epoch 10



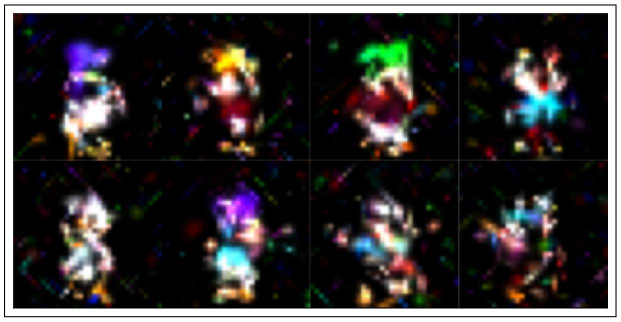
(c) Epoch 20



(d) Epoch 20



(e) Epoch 50



(f) Epoch 50

Figure 8. Comparison of 2×4 sampling grids generated by the pixel diffusion model (left) and the Gaussian diffusion model (right) at epochs 10, 20, and 50. The pixel model shows unstable, noisy samples in early epochs, whereas GaussianDiffusion produces coherent samples from epoch 10 onward.

References

[1] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020. 2

[2] Haisheng Fu, Jie Liang, Zhenman Fang, Jingning Han, Feng Liang, and Guohe Zhang. Weconvene: Learned image compression with wavelet-domain convolution and entropy model. In *European Conference on Computer Vision*, pages 37–53. Springer, 2024. 2

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851, 2020. 1, 2, 4

[4] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. In *ACM Transactions on Graphics (TOG)*, pages 1–12. ACM, 2023. 1, 2

[5] Chenguo Lin, Panwang Pan, Bangbang Yang, Zeming Li, and Yadong Mu. Diffsplat: Repurposing image diffusion models for scalable gaussian splat generation. *arXiv preprint arXiv:2501.16764*, 2025. 2

[6] Wenqian Liu, Runze Li, Meng Zheng, Srikrishna Karanam, Ziyan Wu, Bir Bhanu, Richard J Radke, and Octavia Camps. Towards visually explaining variational autoencoders. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8642–8651, 2020. 2

[7] Akash Malhotra and Nacéra Seghouani. Gaussian splatting: An introduction. *Image Processing On Line*, 2025. 2

[8] William Peebles and Saining Xie. Scalable diffusion models with transformers. *arXiv preprint arXiv:2212.09748*, 2022. 2

[9] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 4

[10] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. *arXiv preprint arXiv:2112.10752*, 2022. 2

[11] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 2, 3

[12] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pages 2256–2265. PMLR, 2015. 1

[13] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 2

[14] Xinjie Zhang, Xingtong Ge, Tongda Xu, Dailan He, Yan Wang, Hongwei Qin, Guo Lu, Jing Geng, and Jun Zhang. Gaussianimage: 1000 fps image representation and compression by 2d gaussian splatting. *arXiv preprint arXiv:2403.08551*, 2024. 1, 2

[15] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 2