LVSM-VAE - Transformer-based novel-view-synthesis in a latent space

Jonas Fischer
TU Munich
jonasetienne.fischer@tum.de

Felix Laarmann
TU Munich
felix.laarmann@tum.de

Abstract

We propose LVSM-VAE, an extension of the Large View Synthesis Model (LVSM) [4] that is adjusted to train within the latent space of a Variational Autoencoder (VAE) to reduce inference cost without compromising quality, enabling the processing of longer image sequences at a given compute budget.

1. Introduction

Novel View Synthesis (NVS) is a long-studied task in computer vision that aims to generate novel perspectives of a scene from reference frames and camera poses. Recent advances have transformed this field through new 3D representations and rendering techniques. NeRF [9] introduced a neural volumetric scene representation to synthesize novel views [3, 5, 14], while 3D Gaussian Splatting (3DGS) [6] proposed a more efficient representation of 3D scenes [2, 12, 16]. In contrast, the Large View Synthesis Model (LVSM) [4] eliminates the need for explicit 3D representations through a transformer-based framework that generates novel views from sparse inputs. However, LVSMs remain highly computationally expensive, requiring up to 64 A100 GPUs for several days of training. In many downstream settings, increasing the number of available reference views directly leads to higher image quality. But it also increases compute demands, making scalability in the number of context images a practical bottleneck. To mitigate this, we propose training LVSM-based models in a Variational Autoencoder (VAE) latent space instead of the high-dimensional pixel space. This approach is inspired by Latent Diffusion Models (LDM) [11], which achieve strong generative performance by operating efficiently within compressed latent representations.

2. Related work

Optimization-based novel view synthesis models reconstruct scenes by optimizing a continuous 3D scene representation for each individual scene from a set of input images, which is then used to render novel views. NeRF [9] introduced the optimization of a volumetric neural radiance field via differentiable rendering, achieving state-of-the-art results in NVS. Gaussian Splatting [6] extends NeRF by representing scenes with explicit 3D Gaussians instead of implicit fields, enabling significantly faster rendering while maintaining comparable visual quality. Numerous extensions build upon NeRF [5, 14] and Gaussian Splatting [12, 16], or explore alternative scene representations such as linear primitives in LinPrim [8].

Learning-based novel view synthesis models eliminate per-scene optimization and instead learn to directly infer a 3D scene representation from a set of input views. PixelNeRF [17] and IBRNet [13] predict volumetric scene representations from sparse context views by leveraging learned 3D priors. PixelSplat [2] extends this line of work by directly regressing 3D Gaussian Splatting representations.

Large Reconstruction Models (LRMs) [3, 15, 19] further scale this paradigm by training transformer-based architectures on large and diverse datasets to acquire strong 3D priors, while still relying on explicit 3D scene representations.

Recently, **LVSM** [4] removed the reliance on explicit 3D scene representations by directly predicting target views in an end-to-end manner in pixel space using a transformer with minimal 3D inductive bias. Subsequent work proposed Projective Positional Encoding (PRoPE) [7], a relative position encoding designed to capture complete camera frustums in multi-view tasks, demonstrating improved performance when integrated with LVSM. While this formulation enables flexible modeling and high-quality novel view synthesis, LVSM remains computationally expensive due to operating in pixel space. This motivates our approach to perform view synthesis in a compressed latent space, inspired by latent generative models such as Stable Diffusion [11].

3. Method

To reduce the computational overhead of pixel-space view synthesis while preserving the flexibility of LVSM, we reformulate the model to operate directly in a compressed latent space.

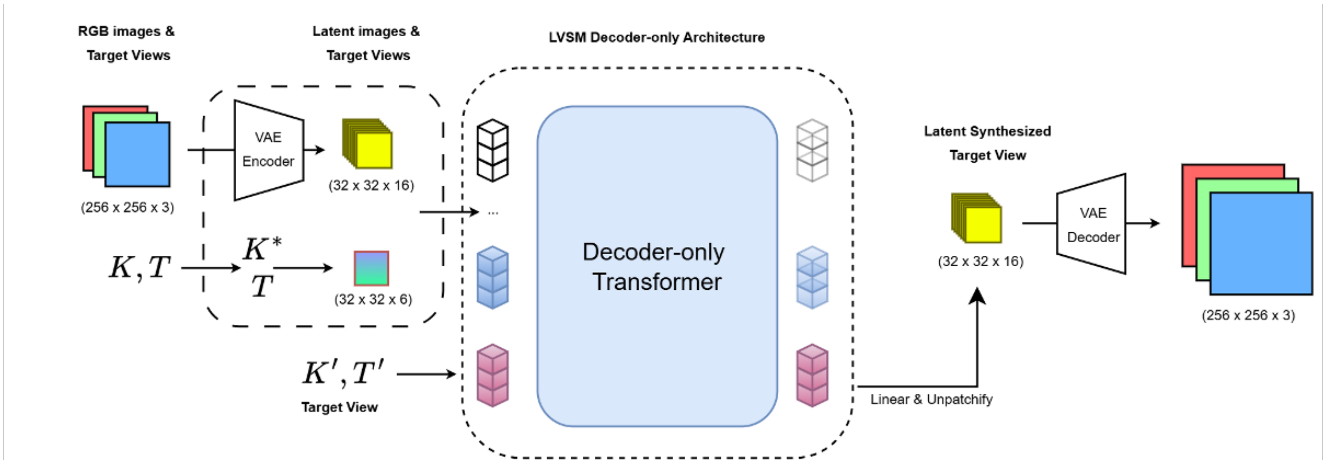


Figure 1. LVSM-VAE architecture. The VAE-Encoder encodes reference RGB image into a latent space and the positional encoding is transformed accordingly to the encoder reduction. Additionally we encode the target view position and give this to our transformer architecture that then predicts the novel target view in latent space. At the end we decode the image again to obtain our target in pixel-space

Our approach first employs a VAE encoder to map the reference images I_i into latent representations I_i^{latent}. The latent feature maps are then patchified with patch size p into $\{I_{i,j}^{\text{latent}} \in \mathbb{R}^{p \times p \times C} \mid j = 1, \dots, HW/(Dp^2)\}$, where D denotes the spatial downsampling factor of the VAE encoder and C the channel dimension of the latent representation.

Before computing pixel-wise ray embeddings, we adapt the camera intrinsic matrix K by scaling the focal lengths and principal point coordinates by the downsampling factor to ensure geometric consistency between pixel space and latent space. Using the adjusted intrinsics K', we compute ray embeddings and similarly patchify them into $\{R_{i,j}^{\text{latent}} \in \mathbb{R}^{p \times p \times C} \mid j = 1, \dots, HW/(Dp^2)\}$. We then follow the Decoder-only LVSM transformer architecture [4] to predict the target latent representation.

The target camera conditioning is represented via ray embeddings R^t, computed from the target camera pose T^t and adapted intrinsic matrix K'^t. Context tokens are constructed by concatenating the latent image patches $I_{i,j}^{\text{latent}}$ with their corresponding ray embeddings $R_{i,j}$ and projecting them through a linear layer. Target ray embeddings are projected independently using a separate linear layer:

$$\mathbf{x}_{i,j} = \text{Linear}_{\text{input}}\big([\mathbf{I}_{i,j}^{\text{latent}}, \mathbf{R}_{i,j}]\big) \in \mathbb{R}^d, \quad (1)$$

$$\mathbf{q}_j = \text{Linear}_{\text{target}}\big(\mathbf{R}_j^t\big) \in \mathbb{R}^d. \quad (2)$$

The model synthesizes the latent representation of the novel view by conditioning the target tokens on the context tokens using attention:

$$y_1, \dots, y_{l_q} = M(q_1, \dots, q_{l_q} \mid x_1, \dots, x_{l_x})[4]. \quad (3)$$

A final linear output layer regresses the latent values of each target patch:

$$\hat{I}_j^{\text{latent},t} = \text{Linear}_{\text{out}}(y_j) \in \mathbb{R}^{Cp^2}. \quad (4)$$

In contrast to the original LVSM formulation, we do not apply a sigmoid activation at the output, as the latent values approximately follow a zero-mean gaussian distribution. Finally, the predicted latent patches are reshaped into their original spatial layout and decoded using the VAE decoder to obtain the synthesized novel view $\hat{I}^t$ in pixel space. The complete architecture can bee seen in Figure 1.

4. Dataset

The RealEstate10k dataset was introduced by Zhou et al. and adopted by many view synthesis models - including the LVSM model and follow-up architectures. It consists of 10 million camera poses and frames extracted from 10,000 YouTube videos of interior scenes. We prepare our training set by following the preprocessing of Li et al. by rescaling and cropping the images to the designated ouput size and afterwards encode them using the VAE.

The dataset is curated in a 90/10 train-test split and an evaluation index mapping three reference views to two target views for a subset of test scenes is provided.

5. Experimental setup

Due to limited computational resources, we adopt a reduced LVSM configuration consisting of 6 transformer blocks and an MLP hidden dimension of 1024, following the setup proposed by Li et al.. For the same reason, all experiments are conducted at an image resolution of 256 × 256.
Batifol et al. introduce Flux, a state-of-the-art image generation model using a convolutional VAE with open weights. We apply their VAE in our pipeline for encoding and decoding.

As training a full pixel-space LVSM baseline under these constraints was infeasible and no pretrained weights were

Method	PSNR ↑	LPIPS ↓	SSIM ↑
LVSM-PRoPE small, 6-layers [7]	22.80	0.146	0.725
LVSM-VAE, 6-layers (ours)	**21.49**	**0.286**	**0.668**

Table 1. Quantitative comparison between LVSM-PROPE and LVSM-VAE using the default RealEstate10k evaluation index.

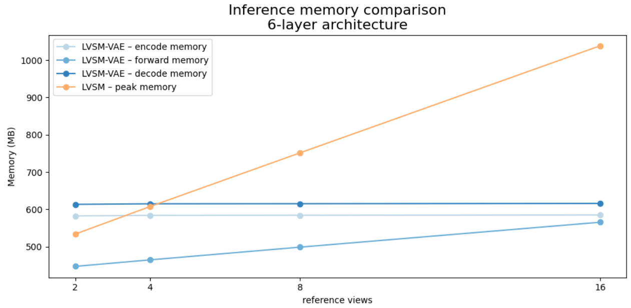


Figure 2. Inference GPU memory comparison between the 6-layer architectures of LVSM-PRoPE and LVSM-VAE. The memory requirements for encoder and decoder remain constant when sequentially encoding reference views and bound the peak memory up to 16 reference views.

publicly available, we instead compare our method against the reported results of Li et al., using an identical model size and evaluation protocol to ensure comparability. All models employ hybrid camera encoding combining PRoPE and CamRay.

We empirically evaluate model performance across varying numbers of input views while training on a fixed number of reference views, assess zero-shot generalization to unseen view counts, and analyze the performance of LVSM-VAE fine-tuned with additional reference views under comparable memory constraints.

6. Results

Although we conducted limited hyperparameter tuning and we stopped training before convergence after 870,000 steps, our LVSM-VAE model achieves competitive results (see Table 1) on the default evaluation index, which measures performance in generating three novel views from two reference images. This result demonstrates that LVSM-based novel view synthesis can be transferred into a latent space with minimal loss.

During inference, GPU memory consumption (Figure 2) and runtime (Figure 3) are dominated by VAE encoding and decoding. However, the memory consumption of the VAE remains constant with respect to the number of reference views, while the transformer forward pass scales with additional inputs and becomes the effective bottleneck. We therefore focus our analysis on the memory consumption of the transformer models. Compared to pixel-space LVSM, LVSM-VAE requires substantially less memory and scales more favorably. Notably, LVSM-VAE with 16 reference views uses less compute than LVSM with four.

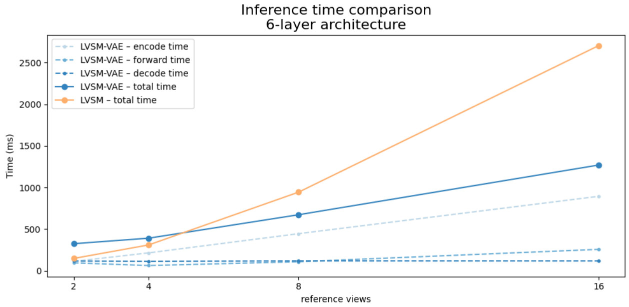


Figure 3. Inference time comparison between the 6-layer architectures of LVSM-PRoPE and LVSM-VAE. For LVSM-VAE the processing time is dominated by sequential encoding that appears to increase linearly with the number of reference views whereas the forward pass time of the LVSM model in pixel space increases exponentially.

Out-of-distribution generalization:. Li et al. report an improvement of PSNR and SSIM values when adapting LVSM using PRoPE to more reference views without any finetuning (Figure 4). Although we apply the PRoPE conditioning as well and we evaluate against the same sampling indexes [7], we find the the performance of our LVSM-VAE model to degenerate when providing with more than two reference images at test time (Figure 4).

Finetuning to more reference views:. Fine-tuning with four reference views improves robustness when providing more than four reference views at inference time. Compared to the original LVSM trained with two reference views, LVSM-VAE fine-tuned on four references achieves higher PSNR while requiring less forward-pass memory (Figure 5), although LPIPS and SSIM remain slightly worse. Further fine-tuning with eight reference views leads to additional gains in PSNR and SSIM compared to the two-references baseline, while maintaining lower memory consumption. LPIPS, however, remains inferior (Figure 5).

7. Discussion

LVSM-VAE scales favorably with the number of reference views, in contrast to the rapidly increasing cost of pixel-space models. This makes the proposed approach particularly suitable for settings with larger context sizes and larger architectures. We observe that generalization to a higher number of reference views depends on the training configuration. Models trained with only two reference views degrade when evaluated with more inputs, whereas fine-

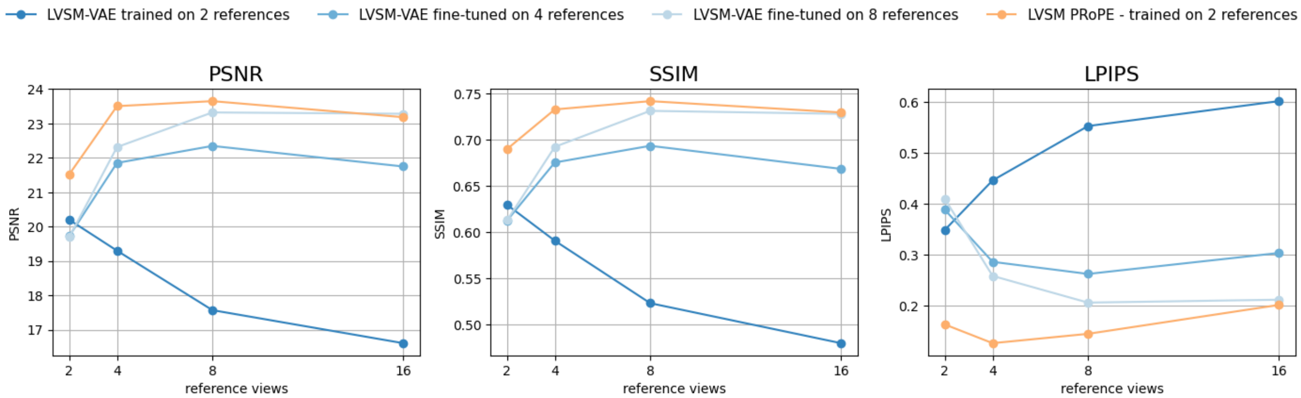


Figure 4. LVSM-VAE inference performance on LVSM-PRoPE context evaluation indexes. When fine-tuning on four or eight references, we observe an improved performance when given eight references.

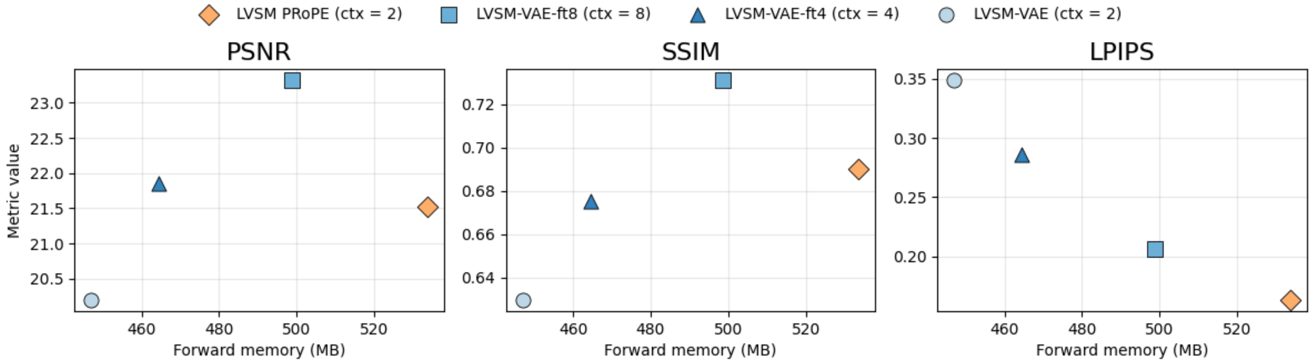


Figure 5. LVSM-VAE inference efficiency on LVSM-PRoPE context evaluation indexes considering fine-tuning with 4 or 8 reference views (ctx). The LVSM-VAE model finetuned and conditioned with eight reference views clearly dominates the LVSM PRoPE model for perceptual metrics, but does not reach the reference performance for LPIPS.

tuning with four reference views improves robustness beyond the training regime. This suggests that LVSM-VAE models benefit from exposure to a wider range of context sizes during training.

When finetuned with eight reference views, our model achieves higher PSNR and SSIM than the pixel-space LVSM baseline, while still exhibiting inferior LPIPS scores. This behavior is expected, as training relies solely on an MSE loss and does not incorporate perceptual loss terms.

8. Conclusion

We find our assumptions validated as we successfully trained an LVSM-model in a VAE latent space and measured significantly reduced runtime and GPU memory requirements during inference. We are able to achieve higher PSNR and SSIM values while using less memory. We note that generalizing to a larger number of reference views required us to train on at least four reference views.

9. Future work

Several directions remain for future work. First, training the proposed model to full convergence may further close the performance gap to pixel-space LVSM. Second, exploring jointly trained encoder–decoder architectures could further accelerate novel view synthesis. Scaling the model to deeper transformer architectures and performing high resolution fine-tuning (as performed by Jin et al.) may further enhance synthesis quality. Additionally, extending the approach to higher image resolutions remains an important direction, particularly with more memory-efficient latent representations. Finally, validating the method on additional datasets beyond RealEstate10k would help assess its generalization capabilities across diverse scenes and capture conditions.

References

[1] Stephen Batifol, Andreas Blattmann, Frederic Boesel, Saksham Consul, Cyril Diagne, Tim Dockhorn, Jack English, Zion English, Patrick Esser, Sumith Kulal, Kyle Lacey, Yam Levi, Cheng Li, Dominik Lorenz, Jonas Müller, Dustin Podell, Robin Rombach, Harry Saini, Axel Sauer, and Luke Smith. FLUX.1 Kontext: Flow Matching for In-Context Image Generation and Editing in Latent Space, 2025. arXiv:2506.15742 [cs]. 2, 1

[2] David Charatan, Sizhe Li, Andrea Tagliasacchi, and Vincent Sitzmann. pixelSplat: 3D Gaussian Splats from Image Pairs for Scalable Generalizable 3D Reconstruction, 2024. arXiv:2312.12337 [cs]. 1

[3] Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. LRM: Large Reconstruction Model for Single Image to 3D, 2024. arXiv:2311.04400 [cs]. 1

[4] Haian Jin, Hanwen Jiang, Hao Tan, Kai Zhang, Sai Bi, Tianyuan Zhang, Fujun Luan, Noah Snavely, and Zexiang Xu. LVSM: A Large View Synthesis Model with Minimal 3D Inductive Bias, 2025. arXiv:2410.17242 [cs]. 1, 2, 4

[5] Mohammad Mahdi Johari, Yann Lepoittevin, and François Fleuret. GeoNeRF: Generalizing NeRF with Geometry Priors. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18344–18347, 2022. 1

[6] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering, 2023. arXiv:2308.04079 [cs]. 1

[7] Ruilong Li, Brent Yi, Junchen Liu, Hang Gao, Yi Ma, and Angjoo Kanazawa. Cameras as Relative Positional Encoding, 2025. Version Number: 1. 1, 2, 3

[8] Nicolas von Lützow and Matthias Nießner. LinPrim: Linear Primitives for Differentiable Volumetric Rendering, 2025. arXiv:2501.16312 [cs]. 1

[9] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, 2020. arXiv:2003.08934 [cs]. 1

[10] William Peebles and Saining Xie. Scalable Diffusion Models with Transformers, 2023. arXiv:2212.09748 [cs]. 1

[11] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-Resolution Image Synthesis with Latent Diffusion Models, 2022. arXiv:2112.10752 [cs]. 1

[12] Stanislaw Szymanowicz, Christian Rupprecht, and Andrea Vedaldi. Splatter Image: Ultra-Fast Single-View 3D Reconstruction, 2024. arXiv:2312.13150 [cs]. 1

[13] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. IBRNet: Learning Multi-View Image-Based Rendering, 2021. arXiv:2102.13090 [cs]. 1

[14] Muyu Xu, Fangneng Zhan, Jiahui Zhang, Yingchen Yu, Xiaoqin Zhang, Christian Theobalt, Ling Shao, and Shijian Lu. WaveNeRF: Wavelet-based Generalizable Neural Radiance Fields, 2023. arXiv:2308.04826 [cs]. 1

[15] Yinghao Xu, Zifan Shi, Wang Yifan, Hansheng Chen, Ceyuan Yang, Sida Peng, Yujun Shen, and Gordon Wetzstein. GRM: Large Gaussian Reconstruction Model for Efficient 3D Reconstruction and Generation. In *Computer Vision – ECCV 2024*, pages 1–20. Springer Nature Switzerland, Cham, 2025. Series Title: Lecture Notes in Computer Science. 1

[16] Botao Ye, Sifei Liu, Haofei Xu, Xueting Li, Marc Pollefeys, Ming-Hsuan Yang, and Songyou Peng. No Pose, No Problem: Surprisingly Simple 3D Gaussian Splats from Sparse Unposed Images, 2024. arXiv:2410.24207 [cs]. 1

[17] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural Radiance Fields from One or Few Images, 2021. arXiv:2012.02190 [cs]. 1

[18] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo Magnification: Learning View Synthesis using Multiplane Images, 2018. arXiv:1805.09817 [cs]. 2

[19] Chen Ziwen, Hao Tan, Kai Zhang, Sai Bi, Fujun Luan, Yicong Hong, Li Fuxin, and Zexiang Xu. Long-LRM: Long-sequence Large Reconstruction Model for Wide-coverage Gaussian Splats, 2025. arXiv:2410.12781 [cs]. 1

LVSM-VAE - Transformer-based novel-view-synthesis in a latent space

Supplementary Material

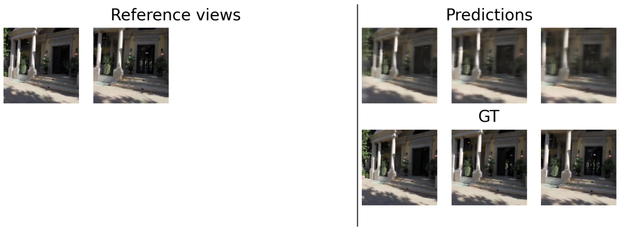


Figure 6. Inferring three predictions on scene f27cb10112666229 with LVSM-VAE trained on two reference views and conditioned on two reference views. SSIM: 0.50

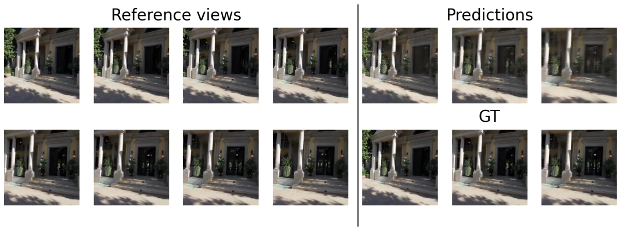


Figure 7. Inferring three predictions on scene f27cb10112666229 with LVSM-VAE trained on two reference views and fine-tuned on four reference views. Conditioned on 8 reference views. SSIM: 0.74

10. Training details

The loss formulation is adapted to the latent space by retaining only the MSE term and omitting perceptual losses. For computational efficiency, training and validation losses are evaluated solely in latent space using MSE. To obtain test metrics, we decode predictions back to the original RGB image space and calculate perceptual metrics (SSIM, LPIPS) in addition to reconstruction error (MSE, PSNR).

To avoid exploding gradients, we reduce the learning rate from originally `4e-4`[4] to `5e-5` with a batch size of 8 and introduce gradient clipping.

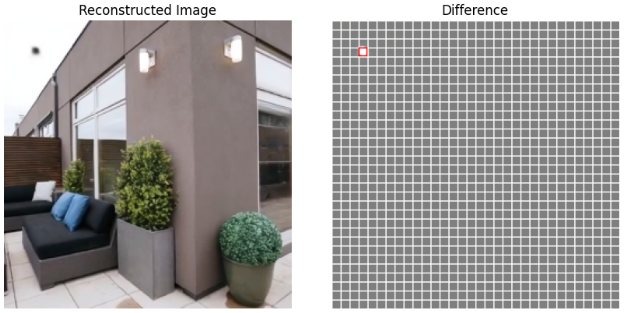
Peebles and Xie evaluate the influence of the token patch size on quality and efficiency when training diffusion models on `32x32` VAE latent images and suggest `2x2` patches as a sweetspot. The Flux.1 architecture follows this advice [1] and so do we.

11. Qualitative Results

We illustrate the effect of increasing the number of reference views on synthesis quality. As shown in Figure 6, using fewer reference views leads to pronounced artifacts and overall blurry reconstructions, whereas Figure 7 demonstrates that providing additional reference views yields noticeably sharper and more coherent predictions.

12. Spatial consistancy study

We explore how the Flux VAE maps spatial relationships from pixel to latent space. Therefore we mask out individual positions across all channels of the latent representations and compare the decoded result to the original image. We observe a strong local effect that stays within or close to the expected grid cell and conclude that beyond rescaling no adjustment of the intrinsics encoding is required.



(a) Decoded image in pixel space. The masked position appears black

(b) Difference between original and decoded image with 8x8 grid.

13. Full scale architecture

While the full-scale LVSM architecture proposed by Jin et al. uses 24 transformer layers with a latent dimension of 3072, Li et al. demonstrated the scalability of PRoPE encodings using a reduced 12-layer model. Building on this setting, we evaluate the computational benefits of latent-space operation by comparing inference memory consumption (Figure 9) and runtime (Figure 10) for 12-layer LVSM models in pixel and latent space. Compared to the 6-layer analysis in Section 6, the advantages of latent-space modeling become more pronounced at this scale, with runtime savings already emerging beyond two reference views.

14. High resolution training

To assess the applicability of our approach at higher image resolutions, we repeat the memory and runtime analysis using `512×512` inputs. While inference-time savings are still observed when more than four reference views are provided (Figure 12), the VAE incurs substantially higher GPU memory consumption during encoding at this resolution (Figure 11). As a result, overall memory savings are only achieved when more than 16 reference views are used.

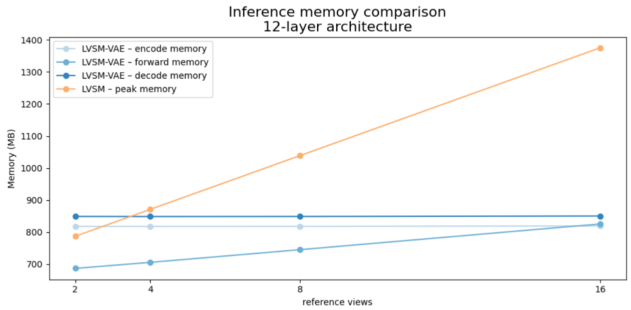


Figure 9. Inference memory comparison of the scaled architecture (12 layers, latent dimension: 3072)

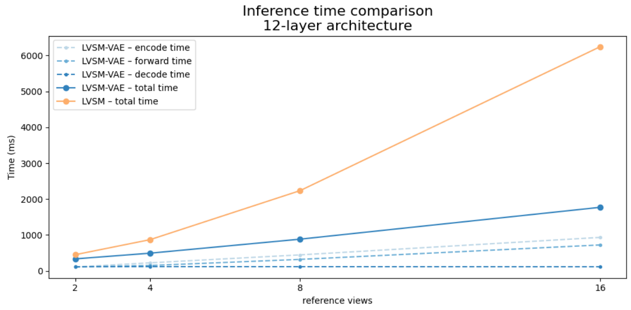


Figure 10. Inference time comparison of the scaled architecture (12 layers, latent dimension: 3072)

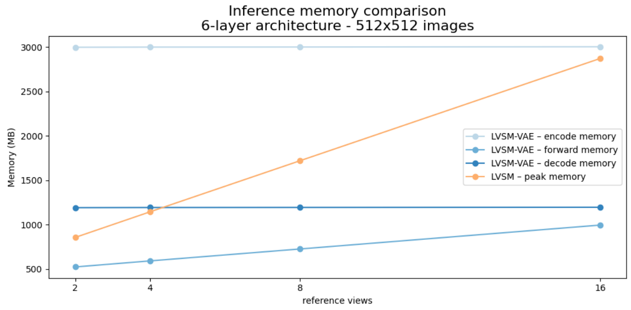


Figure 11. Inference memory comparison when processing 512x512 images

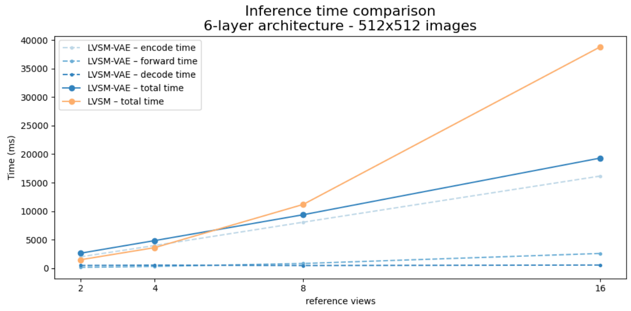


Figure 12. Inference time comparison when processing 512x512 images