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) [5] 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 [10] introduced a neural volumetric scene representation to synthesize novel views [4, 6, 15], while 3D Gaussian Splatting (3DGS) [7] proposed a more efficient representation of 3D scenes [2, 13, 17]. In contrast, the Large View Synthesis Model (LVSM) [5] eliminates the need for explicit 3D representations through a transformer-based framework that generates novel views from sparse inputs. Two variants were proposed: an encoder-decoder architecture that encodes images into a 1D latent scene representation for faster inference, and a decoder-only architecture that directly generates novel views, achieving state-of-the-art quality. Subsequent work proposed Projective Positional Encoding (PRoPE) [8], a relative position encoding designed to capture complete camera frustums in multi-view tasks, demonstrating improved performance when integrated with LVSM. However, LVSMs remain highly computationally expensive, requiring up to 64 A100 GPUs for several days of training.
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) [12], 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 [10] introduced the optimization of a volumetric neural radiance field via differentiable rendering, achieving state-of-the-art results in NVS. Gaussian Splatting [7] 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 [6, 15] and Gaussian Splatting [13, 17], or explore alternative scene representations such as linear primitives in LinPrim [9].

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 [18] and IBRNet [14] 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)** [4, 16, 20] 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** [5] removed the reliance on explicit 3D scene representations by directly predicting target views in a end-to-end manner in pixel space using a transformer with minimal 3D inductive bias. 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 [12].

Variational Autoencoder (VAE) encode RGB images into a latent space of smaller spatial but higher channel

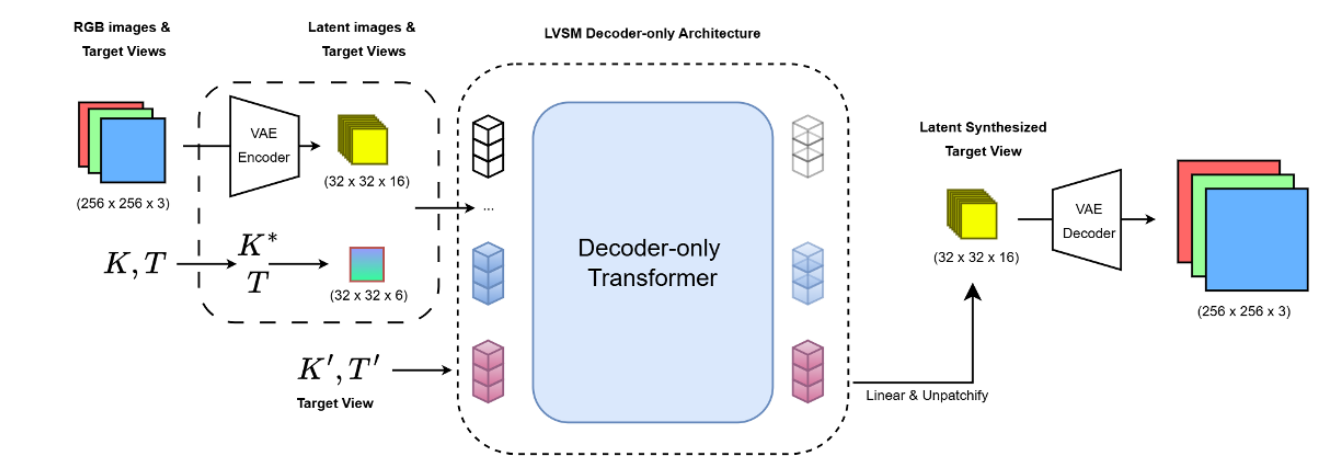


Figure 1. LVSM-VAE architecture. The VAE-Encoder encodes the contex RGB image into latent space and the positional encoding is transformed accordingly to the encoder reduction. Additionally we encode the target view position and give this to our transformer architecture that then predicts the novel target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

dimensionality. Batifol et al. introduce a state-of-the-art image generation model using a convolutional VAE with open weights.

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.

Our approach first employs a VAE encoder to map the context images I_i into latent representations I_i^{latent}, following idea introduced in Stable Diffusion [12]. The latent feature maps are then patchified with patch size p into $\{I_{i,j}^{\text{latent}} \in \mathbb{R}^{p\times p\times C} \mid j = 1, \ldots, HW/(Dp^2)\}$, 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 intrinsic camera 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:

$$K' = \begin{pmatrix} sf_x & 0 & sc_x \\ 0 & sf_y & sc_y \\ 0 & 0 & 1 \end{pmatrix}. \tag{1}$$

Using the adjusted intrinsics, we compute ray embeddings and similarly patchify them into $\{R_{i,j}^{\text{latent}} \in \mathbb{R}^{p\times p\times C} \mid j = 1, \ldots, HW/(Dp^2)\}$. We then follow the end-to-end LVSM transformer architecture [5] to predict the target latent representation.

The target camera pose is represented via ray embeddings R^t, computed from the target extrinsic matrix T^t and 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:

$$\mathbf{x}_{i,j} = \text{Linear}_{\text{input}}\left(\left[\mathbf{I}_{i,j}^{\text{latent}}, \mathbf{R}_{i,j}\right]\right) \in \mathbb{R}^d, \tag{2}$$

$$\mathbf{q}_j = \text{Linear}_{\text{target}}\left(\mathbf{R}_j^t\right) \in \mathbb{R}^d. \tag{3}$$

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

$$y_1, \ldots, y_{l_q} = M(q_1, \ldots, q_{l_q} \mid x_1, \ldots, x_{l_x})[5]. \tag{4}$$

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

$$\hat{I}_j^{\text{latent},t} = \text{Linear}_{\text{out}}(y_j) \in \mathbb{R}^{Cp^2}. \tag{5}$$

In contrast to the original LVSM formulation, we do not apply a sigmoid activation at the output, as the latent values approximately follow a normal 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.

4. Experiments

This section introduces the dataset, describes the experimental setup including how the comparison to the LVSM

baseline is conducted, and concludes with the training details.

4.1. 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're following the preprocessing of Li et al. by rescaling and cropping the images to `256x256` images.

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 - ensuring metrics of approaches validating on this dataset are comparible.

4.2. Experimental setup

By experimentally masking out positions in latent space and comparing the influence on the decoded image, we verify that the latent encodings are spatially consistant with the RGB images (see Section 10). By calculating the reconstruction error of the encoding-decoding process, we verify that the encoding does not introduce a relevant loss of information.

4.3. Training details

The original LVSM architecture normalizes the RGB values to $[0, 1]$ and applies a sigmoid to ensure valid predictions. As we calculated the latent values to approximately follow a normal distribution $\mathcal{N}(0, 2.5)$ we remove this sigmoid. We also adapt the loss formulation to the latent space by retaining the MSE term and removing the perceptual loss. To avoid exploding gradients, we reduce the learning rate from originally `4e-4`[5] 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.

To keep the training efficient, we calculate training and validation metrics solely in the latent space. 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).

5. 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 context 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, while the cost of the LVSM forward pass is substantially reduced. As a result, when more than four reference views are provided, the LVSM-VAE approach becomes efficient, exhibiting nearly constant memory requirements up to 16 reference views.

Method	**PSNR ↑**	**LPIPS ↓**	**SSIM ↑**
LVSM-PRoPE small, 6-layers [8]	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.

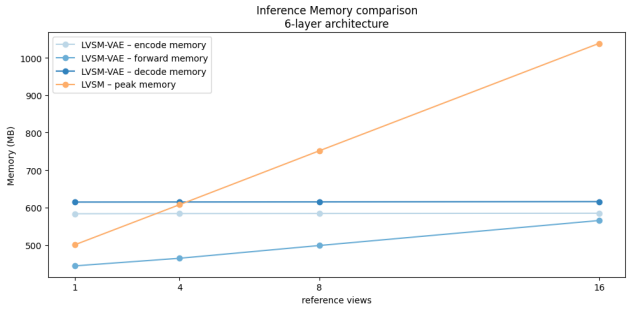


Figure 2. Inference GPU memory comparison between the 6-layer architectures of LVSM-PRoPE and LVSM-VAE. The memory requirements for encoder and decoder remain constant when sequentially encoding reference views and bound the peak memory up to 16 context views.

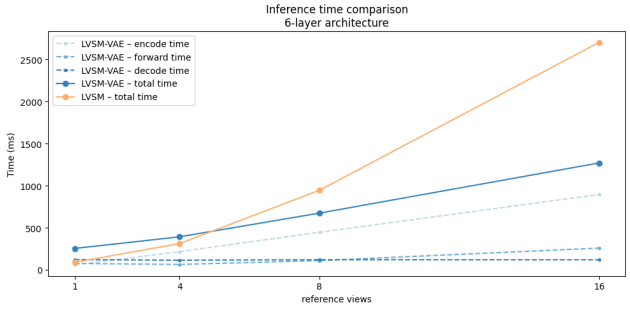
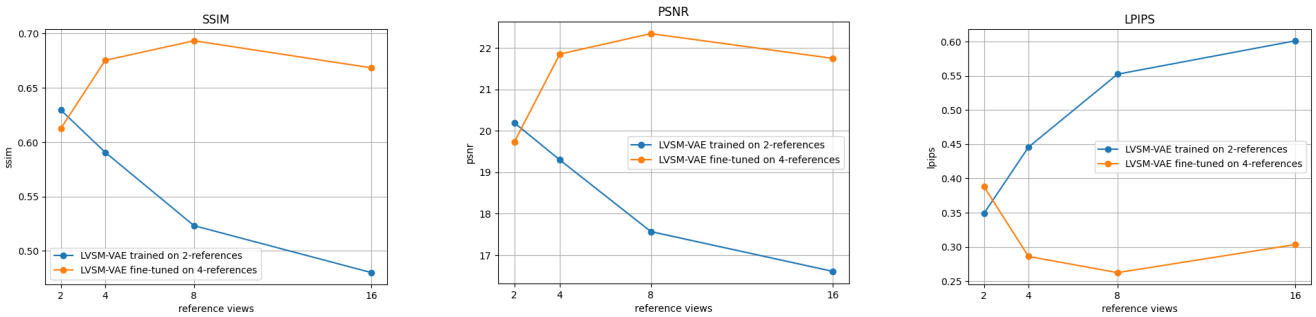


Figure 3. Inference time comparison between the 6-layer architectures of LVSM-PRoPE and LVSM-VAE. For LVSM-VAE the processing time is dominated by sequential encoding that appears to increase linearly with the number of reference views whereas the forward pass time of the LVSM model in pixel space increases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



(a) SSIM (b) PSNR (c) LPIPS

Figure 4. LVSM-VAE inference performance on LVSM-PRoPE context evaluation indexes. Comparison between LVSM-VAE trained on two (blue) versus four (orange) reference images. When training on four references, we observe an improved performance when given eight references.

Jin et al. find LVSM models trained on four input views on the GSO dataset[3] to achieve even higher PSNR values when processing up to 16 input views. Li et al. did not reproduce this effect when training on two input views and the RealEstate10k dataset while applying the original Plücker-based camera conditioning. However, they report a similiar improvement of PSNR and SSIM values when implementing PRoPE camera conditioning. Although we apply the PRoPE conditioning as well and we evaluate against the same sampling indexes [8], 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).

We fine-tune the model with four reference-views and a reduced learning rate of `3e-5` for another 17,000 steps and a batch size of 2. As reported in Figure 4, we observed improved metrics when providing more than two reference views even beyond the trained constellation of four reference views.

6. Discussion

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 note that generalizing to a larger number of reference views required us to train on at least four reference views.

7. 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 note that generalizing to a larger number of reference views required us to train on at least four reference views.

8. Future work

Future work can focus on optimizingTrain to convergence Encoder-Decoder More Layers (full model) und fine-tuning Higher image resolution Validation on more other datasets

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, 3

[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] Laura Downs, Anthony Francis, Nate Koenig, Brandon Kinman, Ryan Hickman, Krista Reymann, Thomas B. McHugh, and Vincent Vanhoucke. Google Scanned Objects: A High-Quality Dataset of 3D Scanned Household Items, 2022. arXiv:2204.11918 [cs]. 4

[4] 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

[5] 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, 3

[6] 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

[7] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler,

and George Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering, 2023. arXiv:2308.04079 [cs]. 1

[8] Ruilong Li, Brent Yi, Junchen Liu, Hang Gao, Yi Ma, and Angjoo Kanazawa. Cameras as Relative Positional Encoding, 2025. Version Number: 1. 1, 3, 4

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

[10] 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

[11] William Peebles and Saining Xie. Scalable Diffusion Models with Transformers, 2023. arXiv:2212.09748 [cs]. 3

[12] 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, 2

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

[14] 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

[15] 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

[16] 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

[17] 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

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

[19] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo Magnification: Learning View Synthesis using Multiplane Images, 2018. arXiv:1805.09817 [cs]. 3

[20] 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

9. Qualitative Results

2-3 scenen aus eval index mit context, target, GT

10. Spatial consistancy study

We explore the how Flux VAE maps spatial relationships from pixel to latent space by masking out individual positions across all channels of the latent representations and compare the decoded result to the

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

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

11. Full scale architecture

12. High resolution training

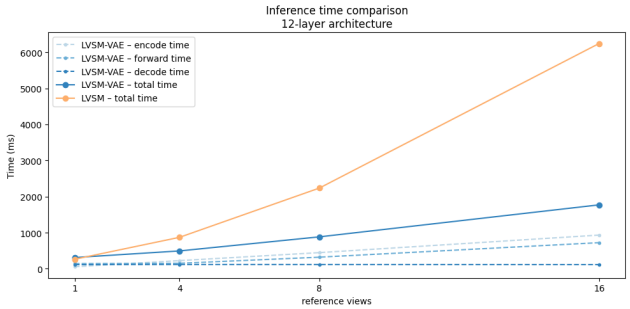


Figure 7. Inference time comparison of the scaled architecture (12 layers, latent dimension: 3072)

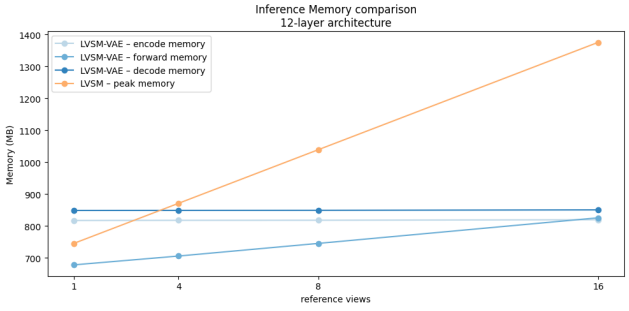


Figure 6. Inference memory comparison of the scaled architecture (12 layers, latent dimension: 3072)