

EFFICIENT CODING AND SYNTHESIS OF GAUSSIAN SPLATTING SCENES

Peter Eisert



About me

Fraunhofer HHI



Vision & Imaging Technologies
Department

Humboldt University Berlin

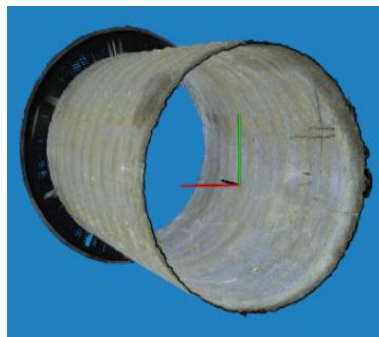


Chair Visual Computing

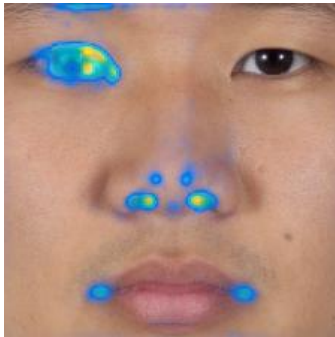


- Visual Computing
- Computer Vision & Graphics
- Deep Learning
- VR/AR/XR
- Applications in multimedia, industry, security, construction, medicine

Research Activities – Deep Vision



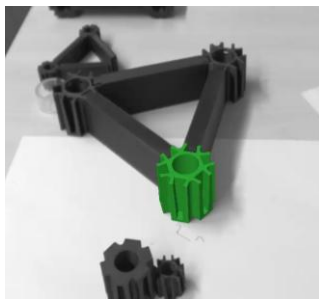
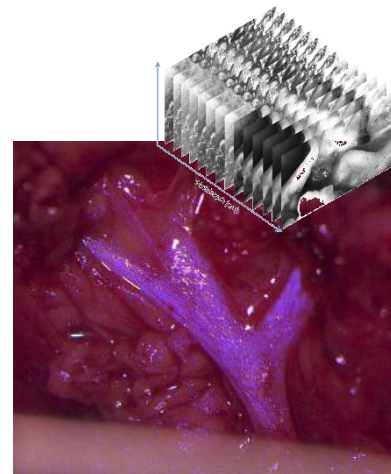
3D Reconstruction



Morph/PAD/Fake Detection



Scene Understanding / Multispectral Imaging



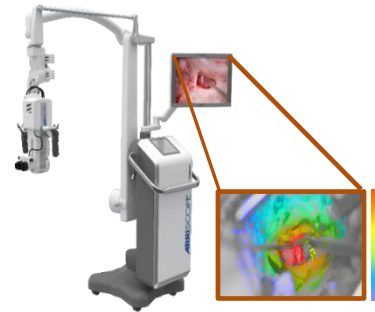
3D Tracking



Analysis and Synthesis of Humans



Augmented / Extended Reality



Fields of Application



Industry & Construction



Medicine



Security



Multimedia



Agriculture & Environment



Mobility

Problem Statement

Reconstructing the 3D World from Images

Input: a set of multi-view images



[...]

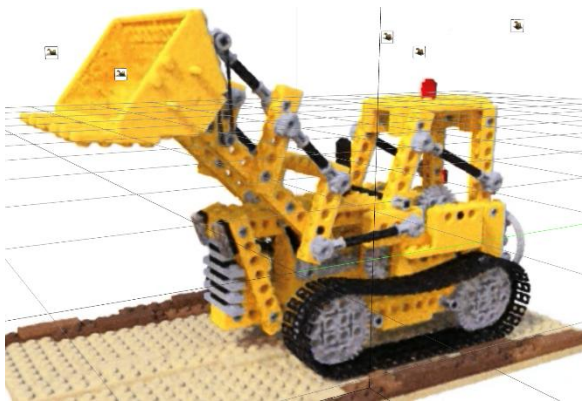


Output: a 3D scene



[3DGS]

Implicit / Neural Representation for 3D Scenes



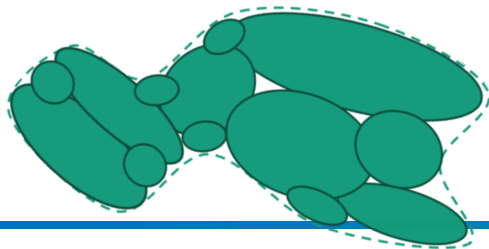
- Neural Radiance Fields (NeRF)
 - implicit representation
 - continuous coordinates
 - color queried from MLP
 - training/rendering quite slow



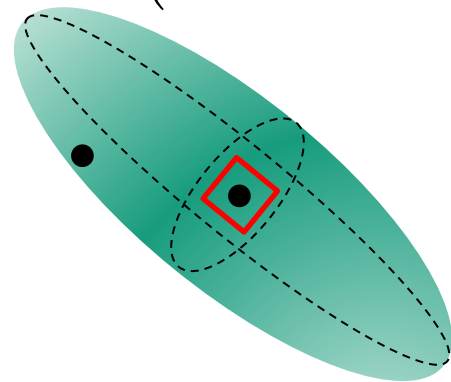
- Gaussian Splatting (3DGS)
 - based on good old point clouds / splatting
 - discrete points, can be moved/edited (4D)
 - each point has many attributes
 - rendering fast

Gaussian Splatting

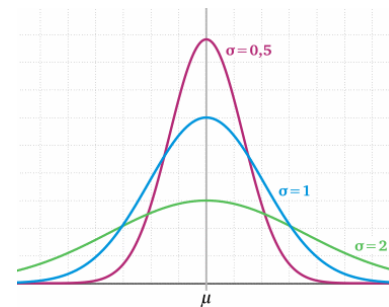
- 3D point cloud with extended, transparent splats
- Splat attributes
 - Mean μ interpretable as location x, y, z
 - Covariance Σ (rotation and anisotropic scale)
 - Opacity $\sigma(\alpha)$
 - Color parameters, 3 values for (R, G, B)
 - Spherical harmonics (SH) for view-dependent colors



$$f_i(\mathbf{p}) = \sigma(\alpha) \exp\left(-\frac{1}{2}(\mathbf{p} - \boldsymbol{\mu}_i) \Sigma_i^{-1} (\mathbf{p} - \boldsymbol{\mu}_i)\right)$$



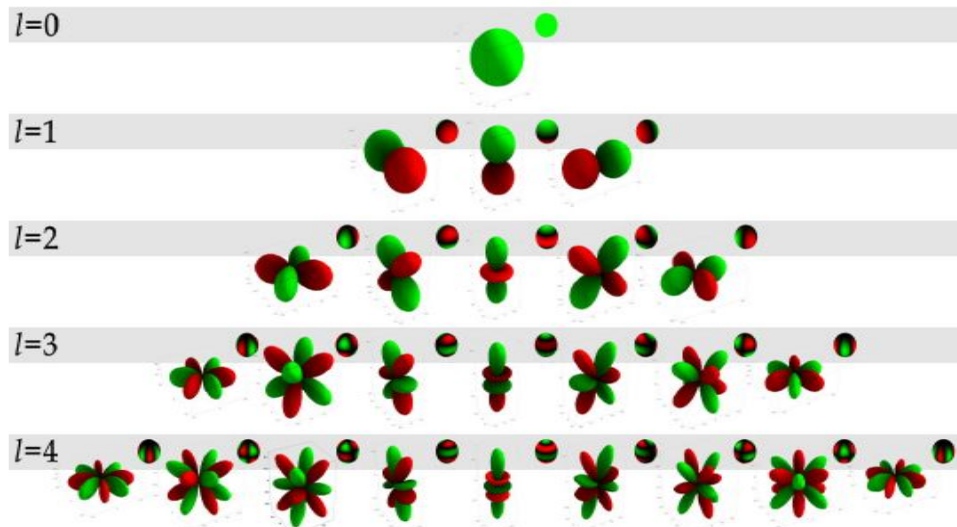
Factorization: $\Sigma = R S S^T R^T$



View Dependency

Spherical Harmonics

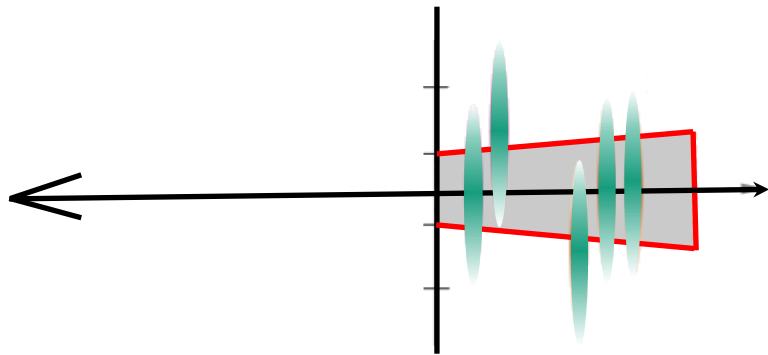
- Representation of view-dependent colors
- Represent directional color and light distribution of each Gaussian
 - Reflection
 - Refraction
 - Specularities
- SH are special functions defined on the surface of a sphere
- Here: order 3
 - → 16 params per color channel



$$Y_l^m(\theta, \phi) = \frac{(-1)^l}{2^l l!} \sqrt{\frac{(2l+1)(l+m)!}{4\pi(l-m)!}} e^{im\phi} (\sin \theta)^{-m} \frac{d^{l-m}}{d(\cos \theta)^{l-m}} (\sin \theta)^{2l}$$



Rendering of GS Representations



$$c(\mathbf{p}) = \sum_{i \in N} c_i \alpha'_i(\mathbf{p}) \prod_{j=1}^{i-1} (1 - \alpha'_j(\mathbf{p}))$$

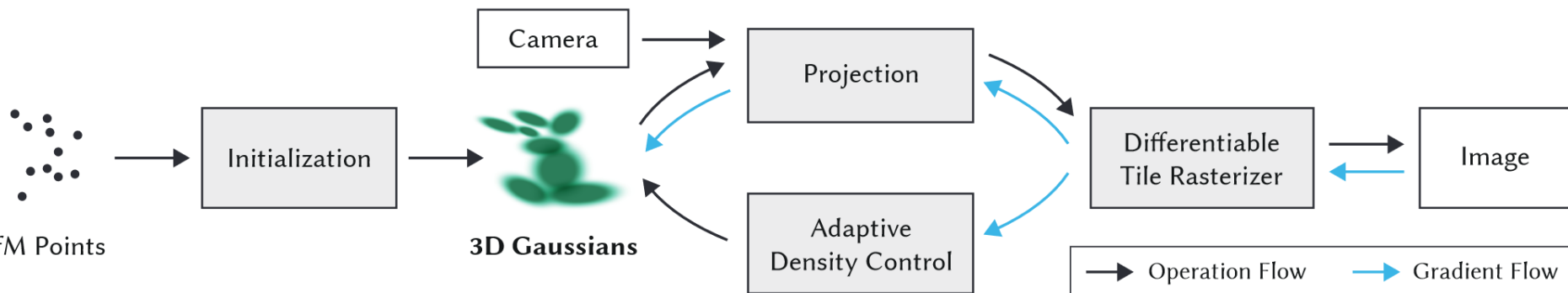
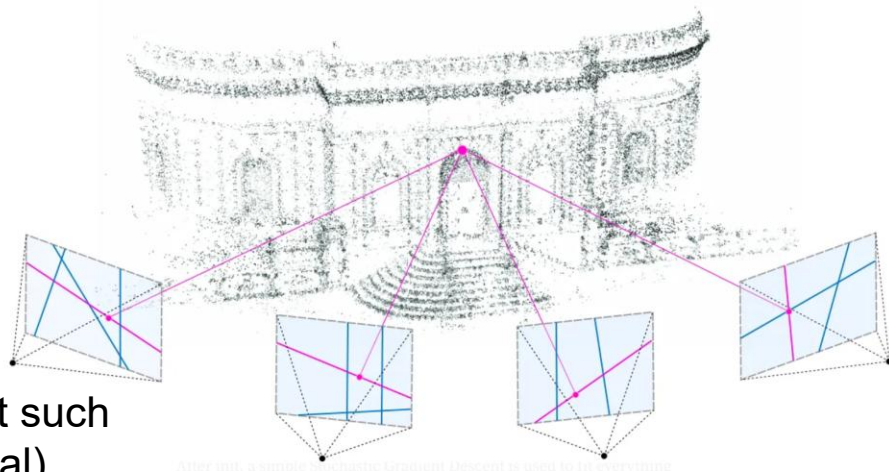
accumulated transmittance

- 3D Gaussians projected on 2D plane parallel to image plane -> 2D splat $\Sigma^{2D} = J W \Sigma W^T J^T$
- Rendered with regular rasterizer; screen is split into 16x16 tiles
- All splats visible in one tile are sorted according to depth (fast radix sort)
- Rendering is then performed from front to back; stopping at a certain alpha saturation value



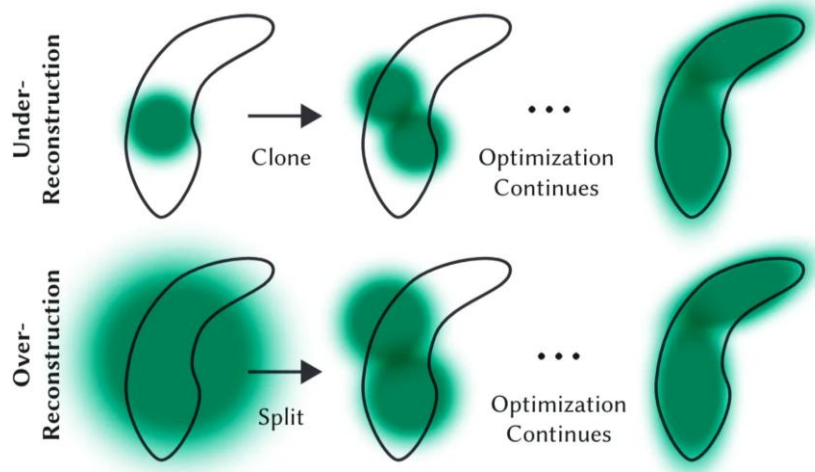
Training of GS Scenes

- Initialize point cloud from SfM
- Differentiable renderer renders images for each provided view and compare with original
- Optimize GS parameters via Gradient Descent such that rendered images improve (closer to original)
- Adaptive densifications: add or remove splats in regular intervals



Densification

- Gradient-based optimization changes attributes of Gaussians but not their number
- In regular intervals, Gaussians are inserted or deleted by 3 measures
 - Cloning (create 2 Gaussians with similar attributes)
 - In areas, where details cannot be reproduced, a Gaussian is cloned
 - Splitting (split a large Gaussian into 2 smaller ones)
 - If a Gaussian gets too large, it is split into 2 smaller ones, placed along the gradient
 - Pruning (deletion of useless Gaussians)
 - Remove Gaussian if alpha is below a threshold (almost invisible)





Kefermarkt Church

By Schindler3D in cooperation with
Fotomento

A late Gothic gem built around 1470,
famous for its extraordinary 15th-century
winged altar. A unique blend of sacred
art, history and living faith in the heart of
Upper Austria.

Use WASD keys to move and mouse to look.

Relighting of Gaussian Splatting Scenes

[Corona et al. R3GW: Relightable 3D Gaussians for Outdoor Scenes in the Wild, VISAPP 2026]

- Lighting typically baked into colors
- Explicit modelling of lighting via environment maps and reflection via BRDFs
- Enables relighting and reconstruction from in the wild images



GS Editing

Prompt-based Editing of 3DGS Scenes

Rendered novel views of a 3DGS scene before (left) and after editing (right)

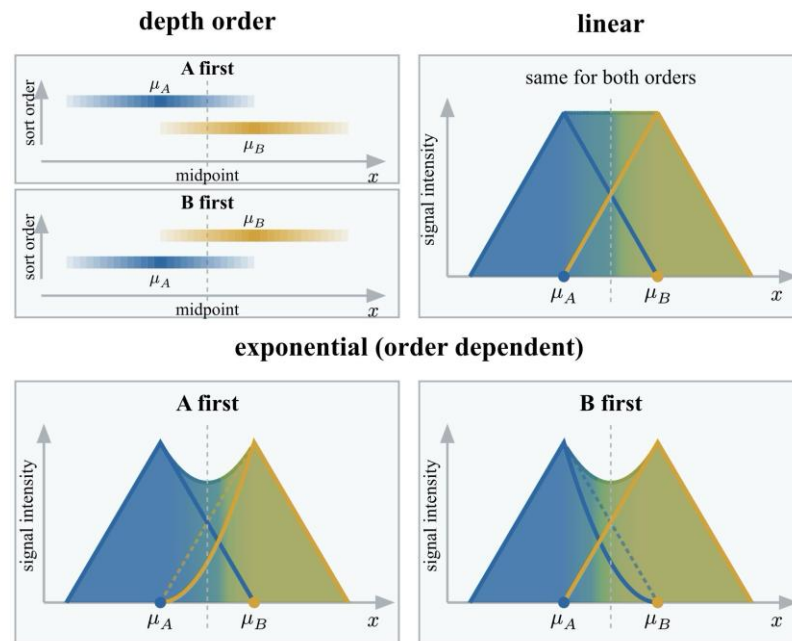


“Turn the bear into a golden bear”



Linear Transmittance for Neural Shaders

- GS sorted according to center
 - popping artifacts
- Linear instead of exponential weights
 - less artifacts
- Triangle primitive with neural shader
 - disentangle geometry from texture
 - decoder can correct imperfections

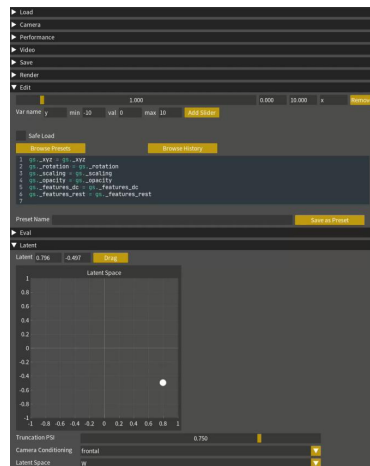


Efficient Coding



[Morgenstern et al. **KISS-GS: 3D Gaussian Splatting Compression Kept Simple**, ECCV 2026]

GAN based Synthesis



[Barthel et al. **COSY: 3D Gaussian Splatting Compression Kept Simple**, ECCV 2026]

Gaussian Splatting

Photorealistic Appearance but Huge in Size

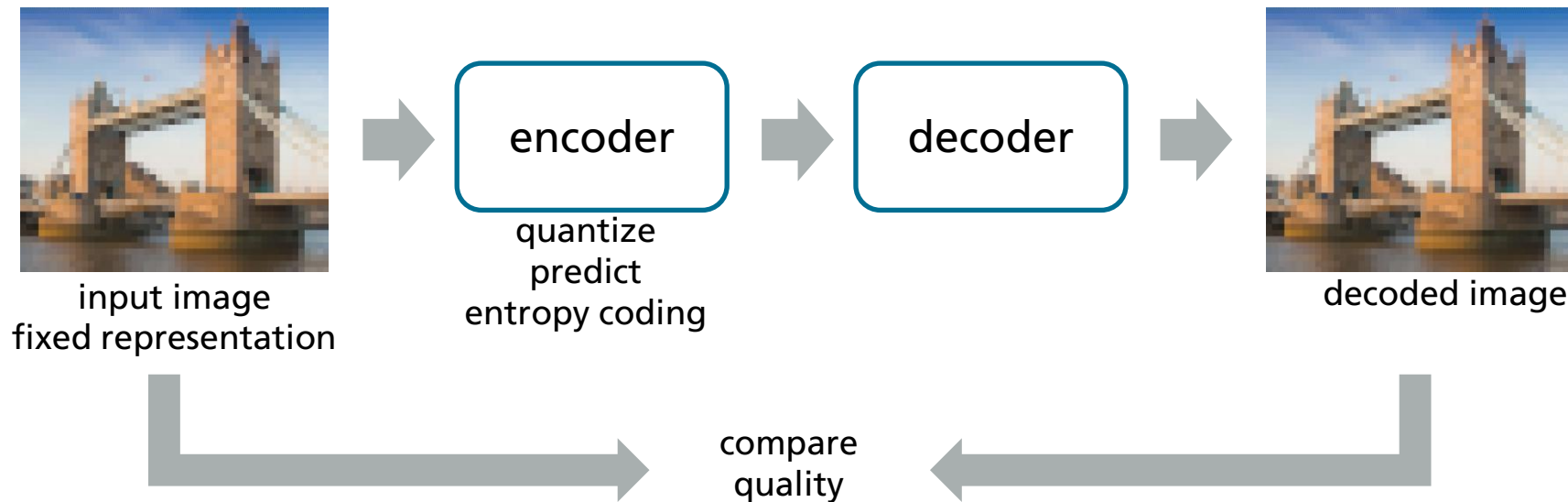


$5,834,784 \text{ [splats]} * 59 \text{ [attributes]} * 4 \text{ [float 32]} = 1,34 \text{ GB}$

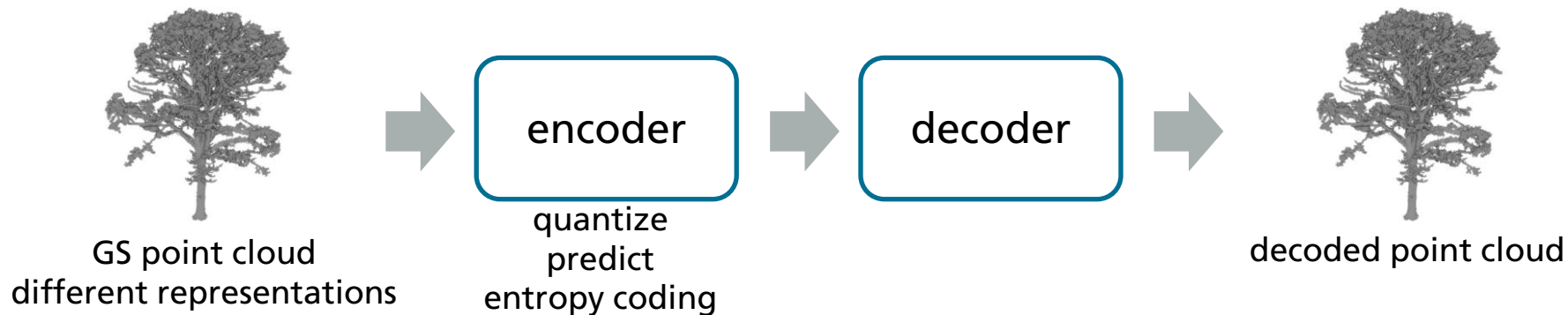
- Parameters of a Gaussian:
 - 3 coordinates – xyz
 - 3 scales
 - 4 rotation (quaternion)
 - 1 opacity
 - 3 RGB color (DC SH)
 - 45 Spherical Harmonics (view dependent effects)
- $14 + 45 = 59$ parameters

Compression of Gaussian Splats

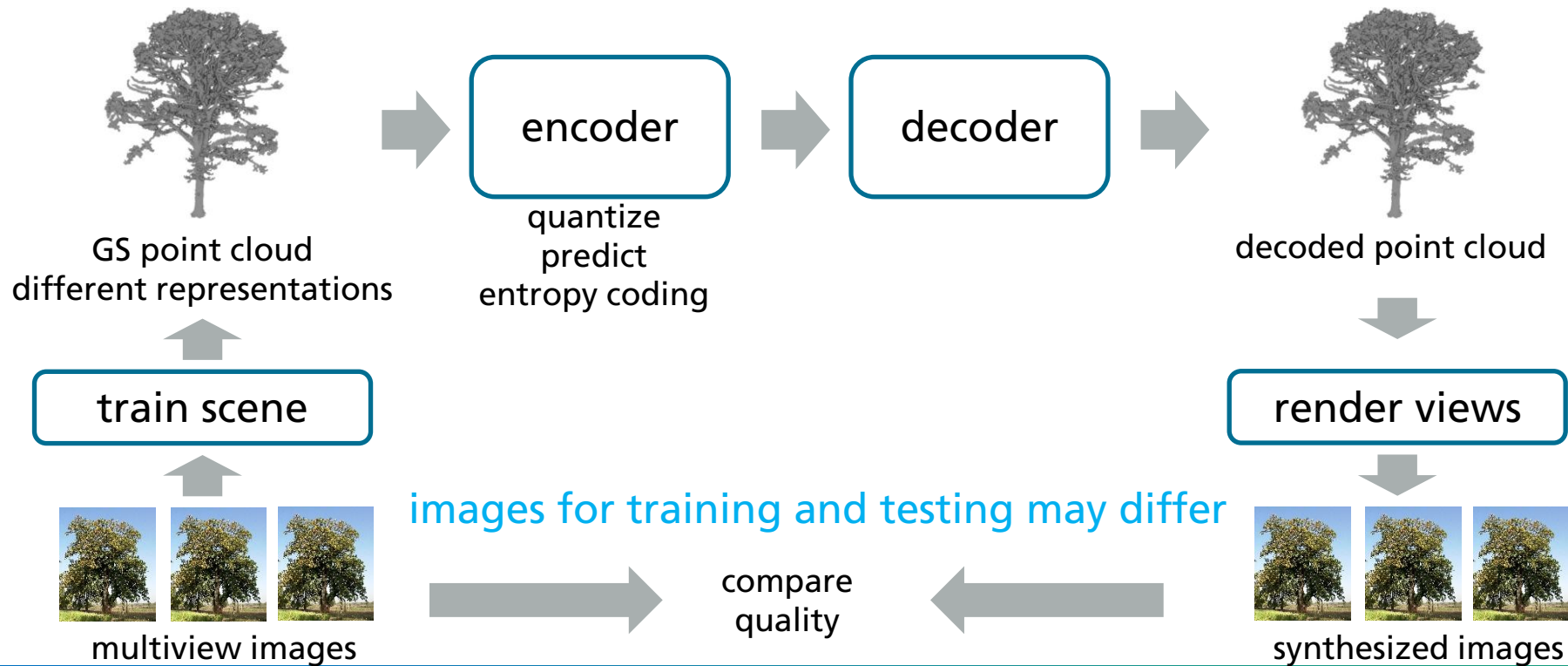
Comparison with Image Coding



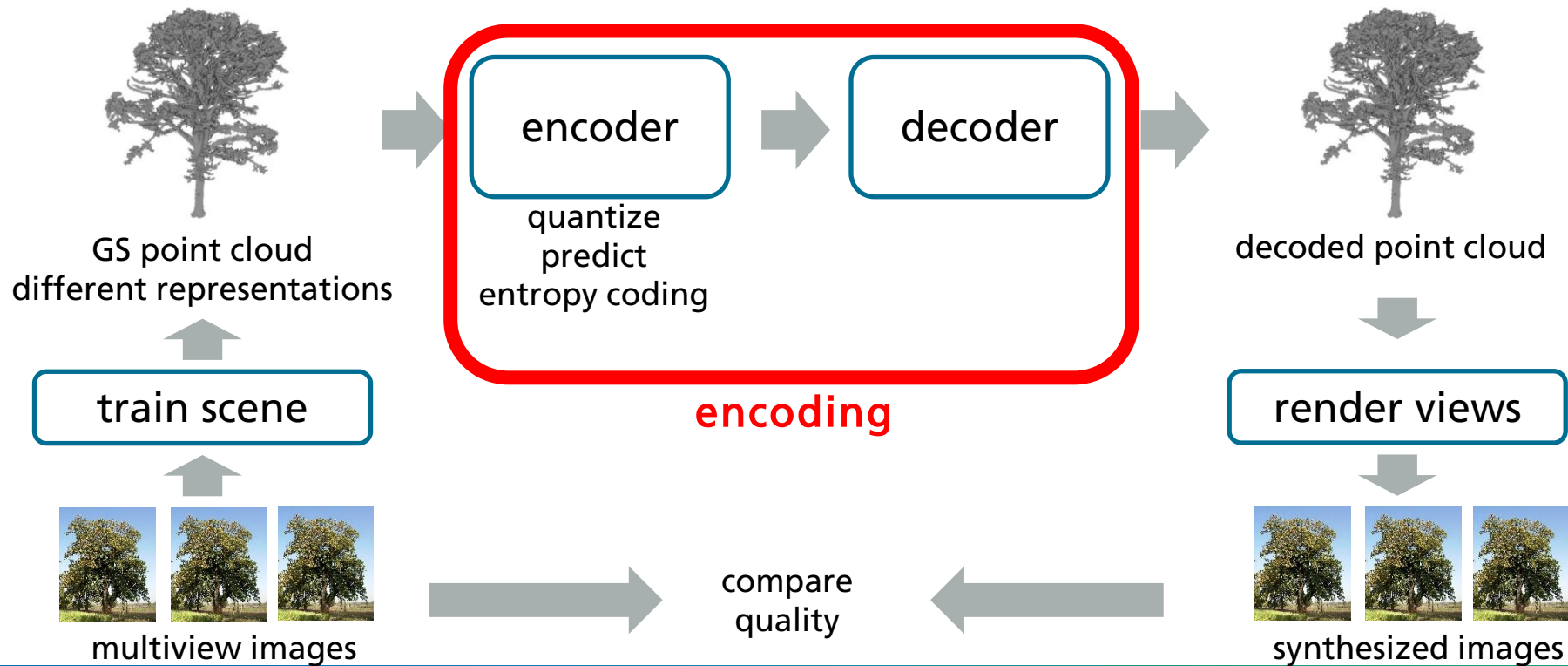
Compression of Gaussian Splats



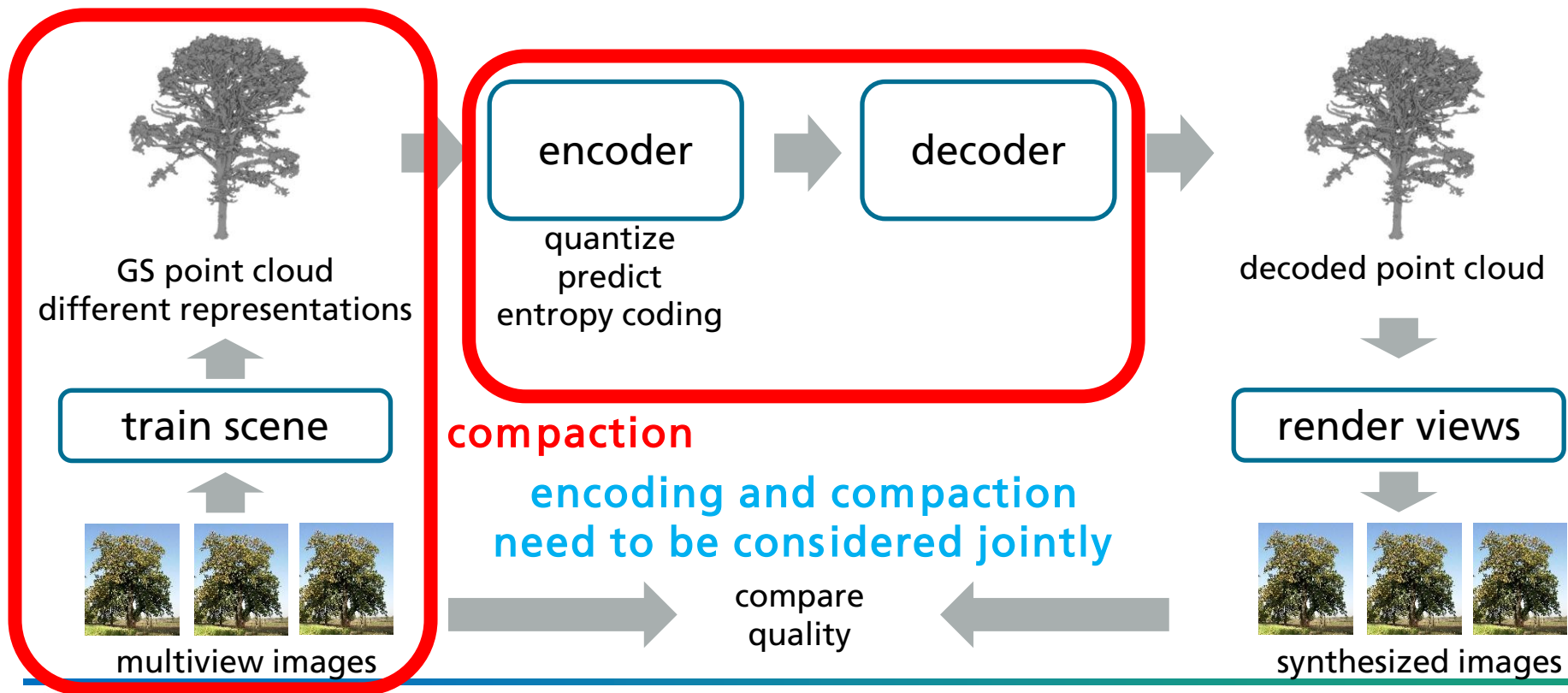
Compression of Gaussian Splats



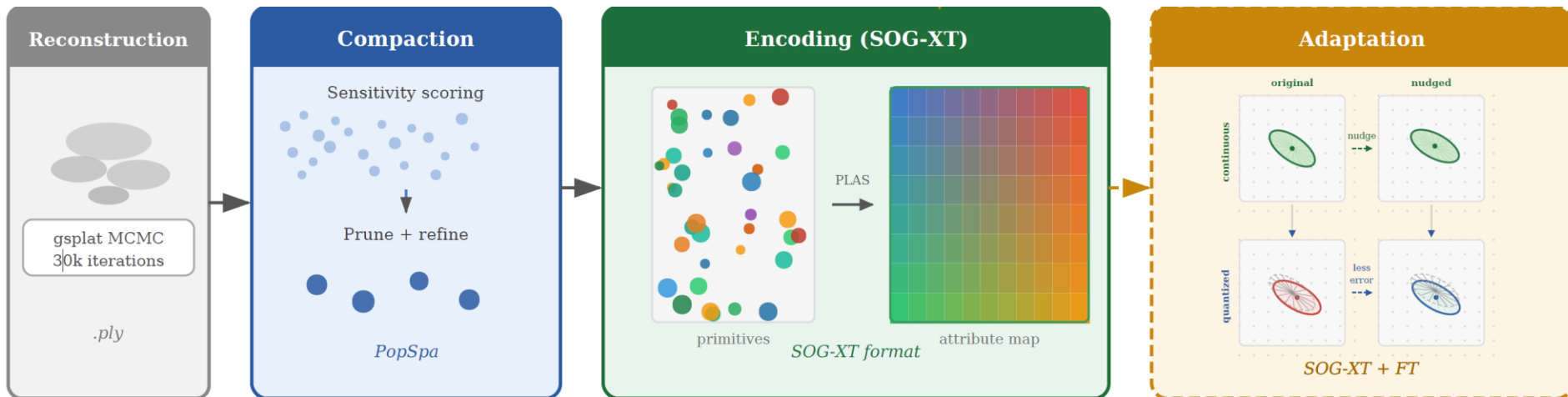
Compression vs. Compaction



Compression vs. Compaction



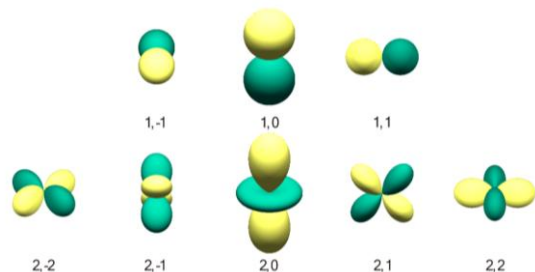
3 Ways of Reducing 3DGS File Sizes



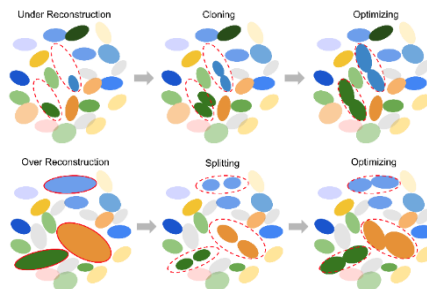
- Compaction, encoding, adaptation
- Typically, all or most strategies are applied jointly
- Difficult to compare approaches

Compaction (Modify Representation to be Coded)

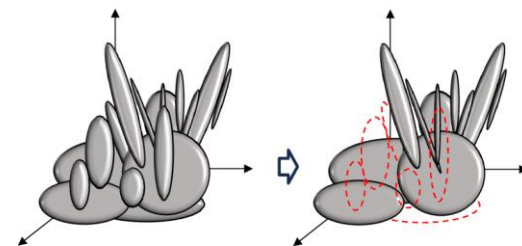
Different Strategies to Get Smaller GS Representations



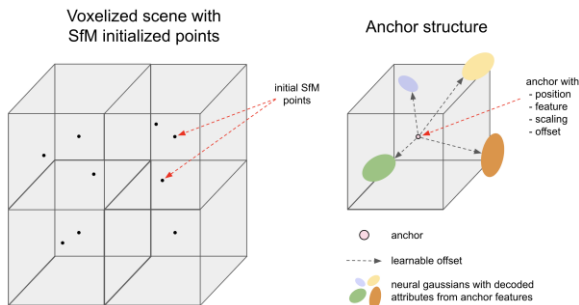
different parameterization



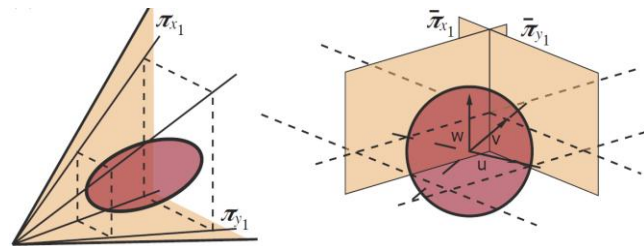
densification & pruning



N Gaussians
Masked Gaussians
coding aware training



structural changes



efficient rendering

Compact 3D Scene Representation via Self-Organizing Gaussian Grids

ECCV 2024



Wieland Morgenstern¹, Florian Barthel^{1,2}, Anna Hilsmann¹, Peter Eisert^{1,2}

¹Fraunhofer Heinrich Hertz Institute, HHI

²Humboldt University of Berlin

Paper

Code

Poster

ECCV 2024 Redux Talk

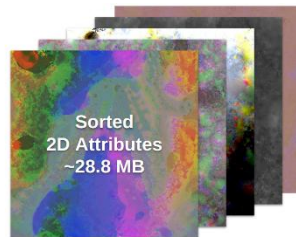
View Dependent Podcast



Millions of Gaussians ~640 MB					
Index	Position	Color	Scale	Opacity	
0	(0.146, 0.401, 0.294)	(0.871, 0.017, 0.886)	(0.499, 0.408, 0.471)	0.707	...
1	(0.481, 0.319, 0.554)	(0.363, 0.796, 0.279)	(0.576, 0.033, 0.913)	0.912	...
2	(0.779, 0.833, 0.399)	(0.207, 0.144, 0.421)	(0.493, 0.581, 0.972)	0.574	...
3	(0.538, 0.205, 0.549)	(0.470, 0.403, 0.837)	(0.192, 0.458, 0.771)	0.467	...
4	(0.135, 0.054, 0.809)	(0.308, 0.882, 0.805)	(0.394, 0.532, 0.026)	0.503	...
5	(0.148, 0.601, 0.295)	(0.871, 0.017, 0.886)	(0.499, 0.408, 0.471)	0.702	...
...	...	...	...	...	...
2.5e7	(0.027, 0.134, 0.032)	(0.204, 0.565, 0.810)	(0.877, 0.552, 0.414)	0.587	...

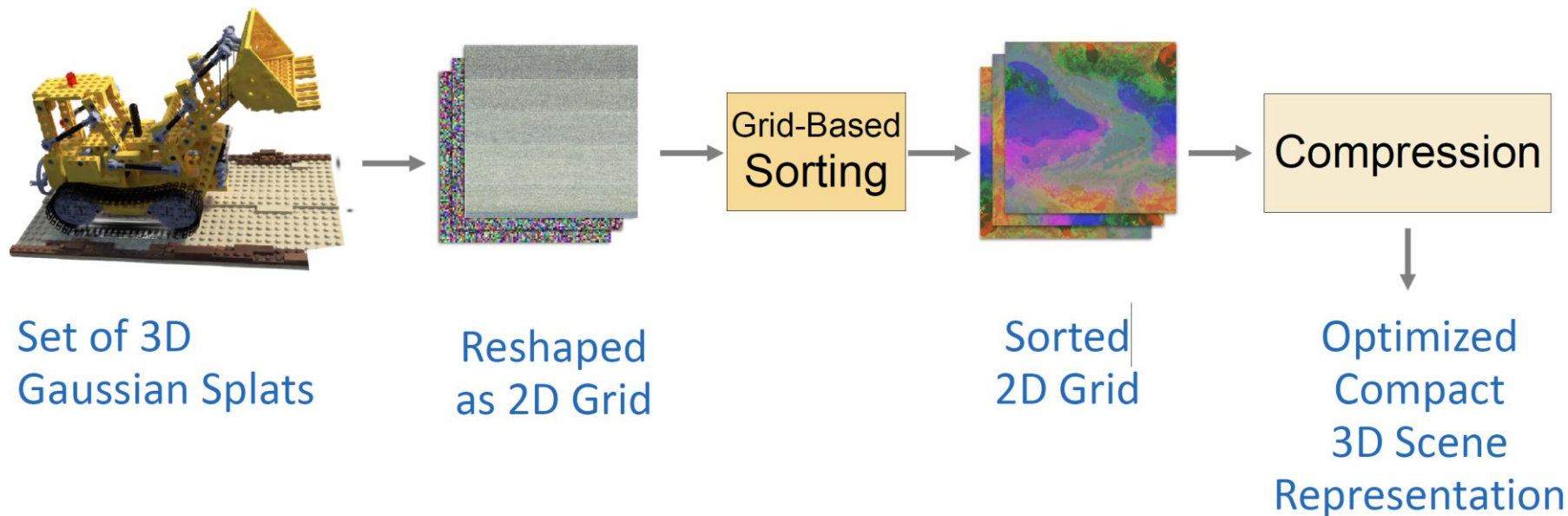


PSNR 25.44dB

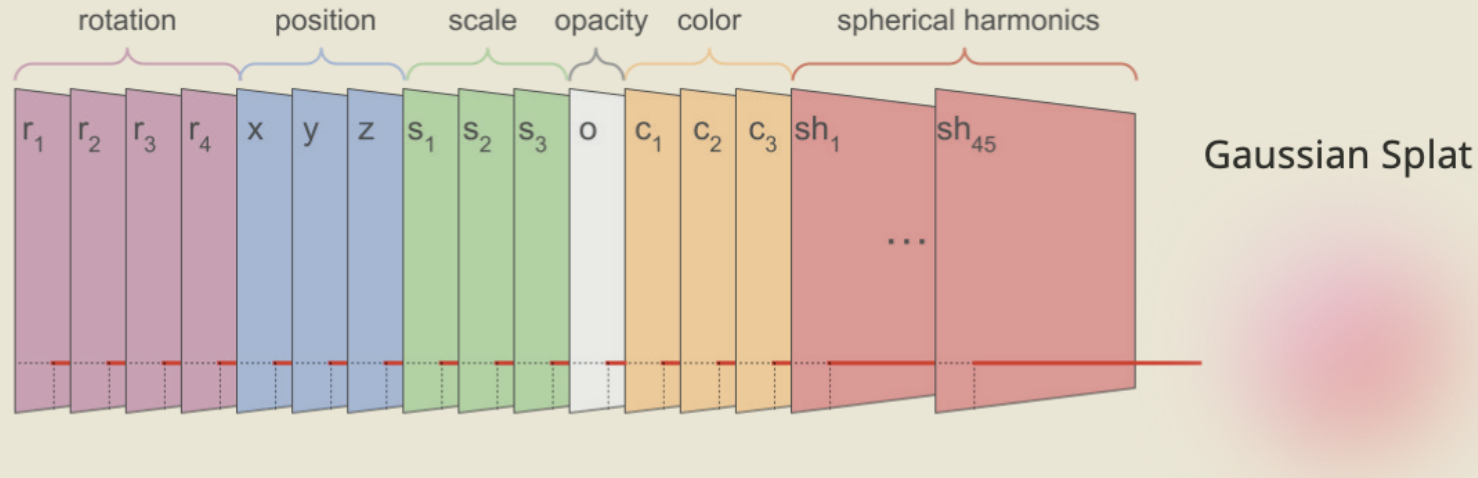


PSNR 25.57dB

Core Concept of Self Organizing Gaussians



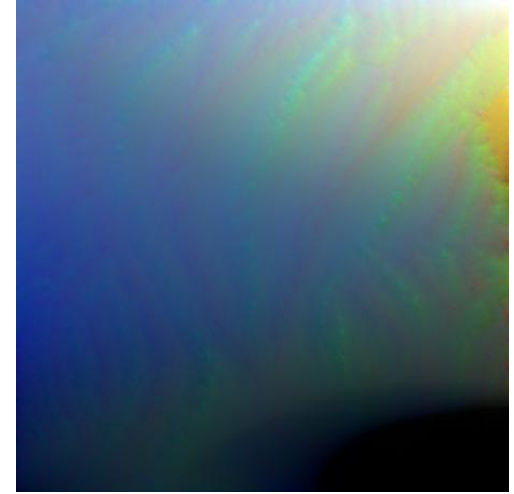
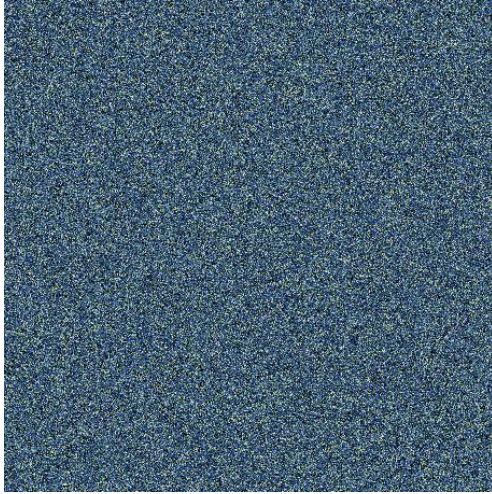
Grids of 3DGS attributes



A single Gaussian splat needs 59 attributes to render.

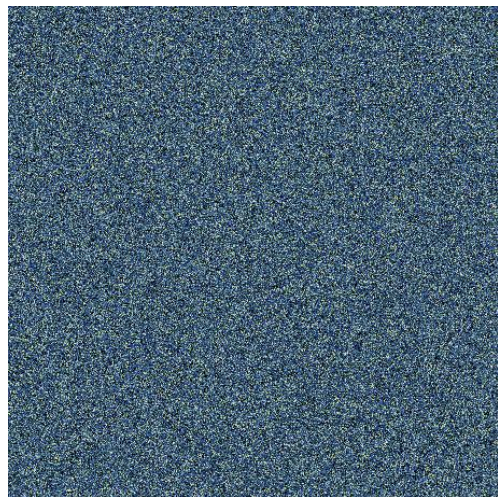
We can reshape the attributes of a scene of n Gaussians into a grid of shape $\sqrt{n} \times \sqrt{n} \times 59$

Compressing sorted data



Compressing sorted data

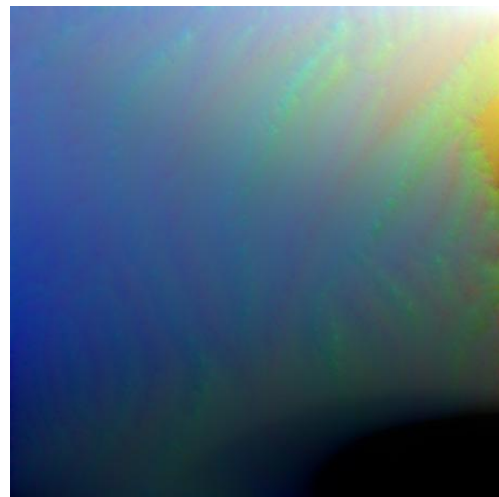
PNG – lossless compression



626 KB



496 KB



199 KB

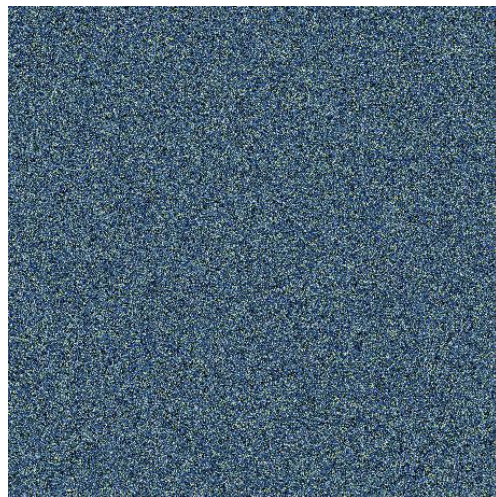
1.26x reduction

2.49x reduction

3.15x reduction

Compressing sorted data

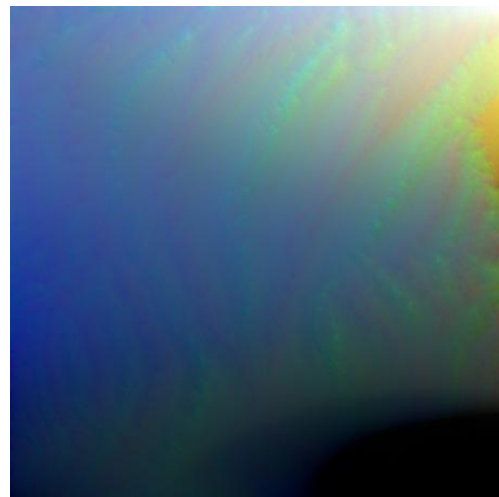
JPEG – lossy compression



298 KB



174 KB



33 KB

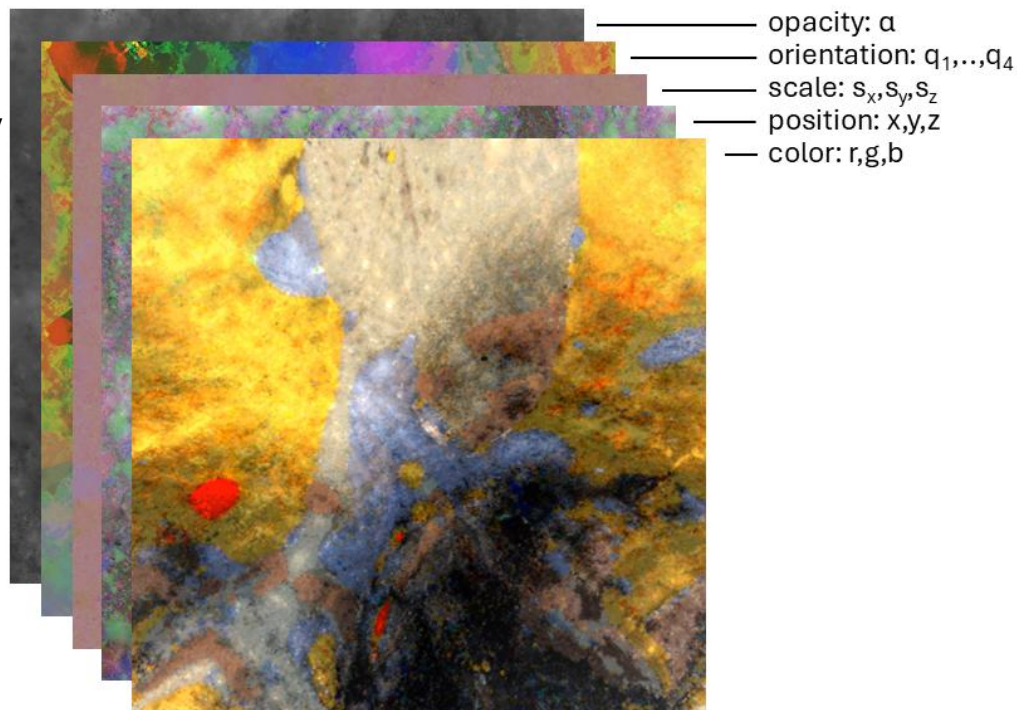
1.71x reduction

5.27x reduction

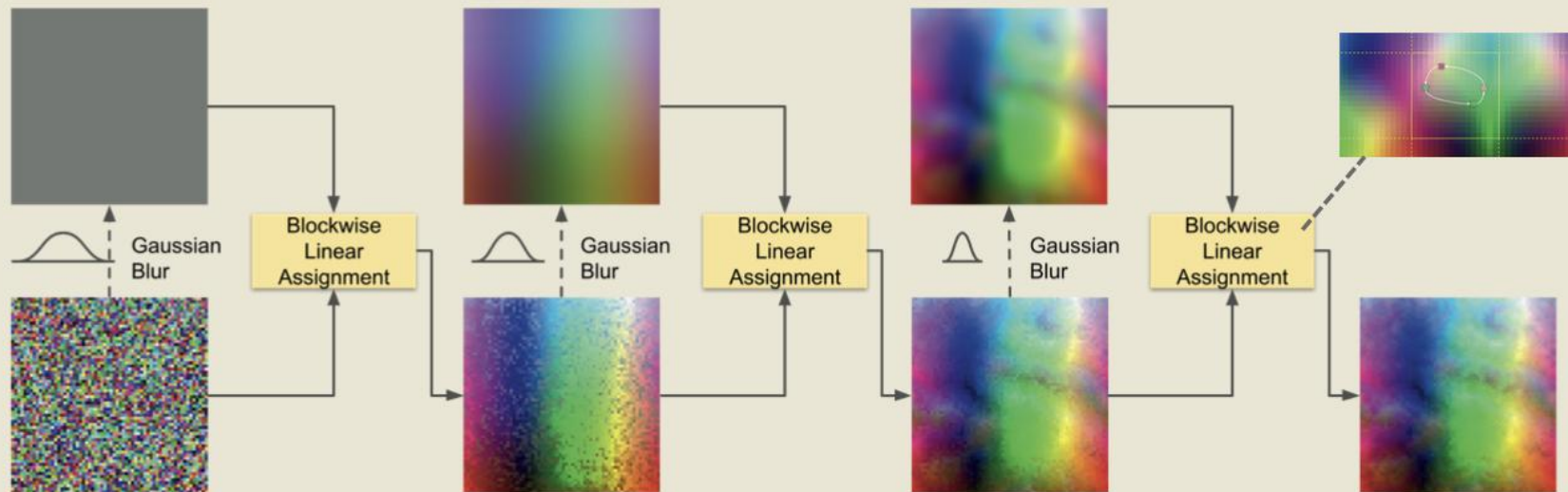
9.03x reduction

Costs for Sorting

- Each Gaussian has several attributes
- All attributes A_i must be sorted identically
- One cost function used for sorting:
 - $c = \sum_i \omega_i \cdot A_i$
- Must be balanced across attributes
- Position very important
(and correlated with other attributes)



Parallel Linear Assignment Sorting

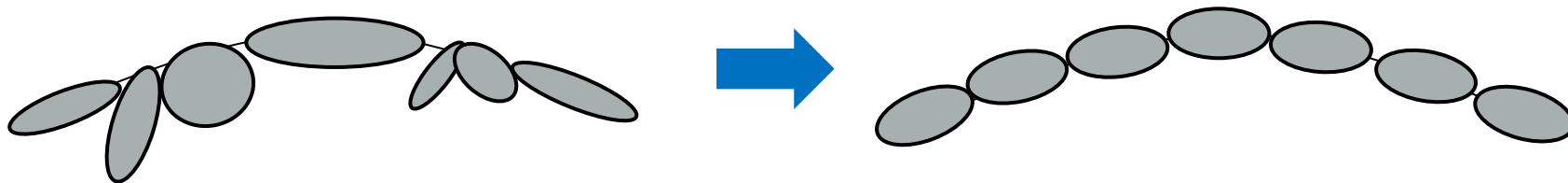


Our novel algorithm: Sorting multi-dimensional data into locally smooth 2D grids. Allows us to group 3D Gaussian Splats with similar attributes close together in 2D.

Self Organizing Gaussians [SOG]



Adaptation



- Gaussian Splatting scenes exhibit many ambiguities to create the same scene
- Optionally, a finetuning can be conducted after reconstruction
 - additional loss: smoothness in attribute planes
 - can further reduce bit-rate without drop in visual quality

Quantization

■ Quantization of attributes

- Position 16 bit can be split into high byte low byte
- Color 8 bit
- Scale, Rotation 6 bit
- Opacity 6 bit
- Spher. Harmonics 5 bit

■ Compress 59 grids as 2D images

- using JPEG XL or WebP with different parameters, one file per attribute group
- spherical harmonics: an image with 15 layers

Results

- Size reduction by a factor of 20-40
- No structural modification necessary
- Supports hardware decoding
- Easy to extend to new research trends
- Basis of most methods within MPEG

3D Gaussian Splatting

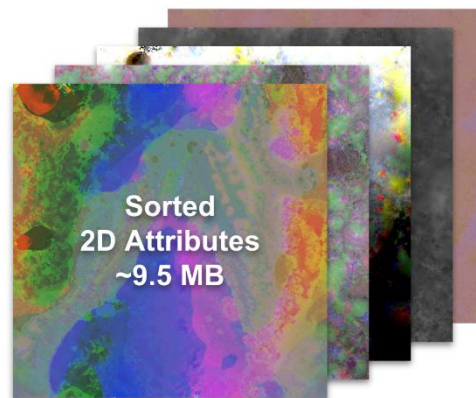
Millions of Gaussians
~174 MB

Index	Position	Color	Scale	Opacity	...
0	(0.148, 0.601, 0.294)	(0.871, 0.917, 0.886)	(0.499, 0.639, 0.471)	0.722	...
1	(0.481, 0.319, 0.554)	(0.983, 0.799, 0.379)	(0.576, 0.653, 0.913)	0.912	...
2	(0.773, 0.833, 0.396)	(0.207, 0.164, 0.421)	(0.693, 0.551, 0.672)	0.074	...
3	(0.838, 0.267, 0.669)	(0.475, 0.478, 0.521)	(0.192, 0.459, 0.771)	0.667	...
4	(0.135, 0.054, 0.808)	(0.308, 0.882, 0.829)	(0.394, 0.532, 0.026)	0.503	...
5	(0.148, 0.601, 0.295)	(0.871, 0.917, 0.886)	(0.499, 0.639, 0.471)	0.722	...
...	...	...	...	...	...
2.5e7	(0.027, 0.134, 0.02)	(0.204, 0.669, 0.816)	(0.877, 0.553, 0.416)	0.587	...



PSNR 24.89dB

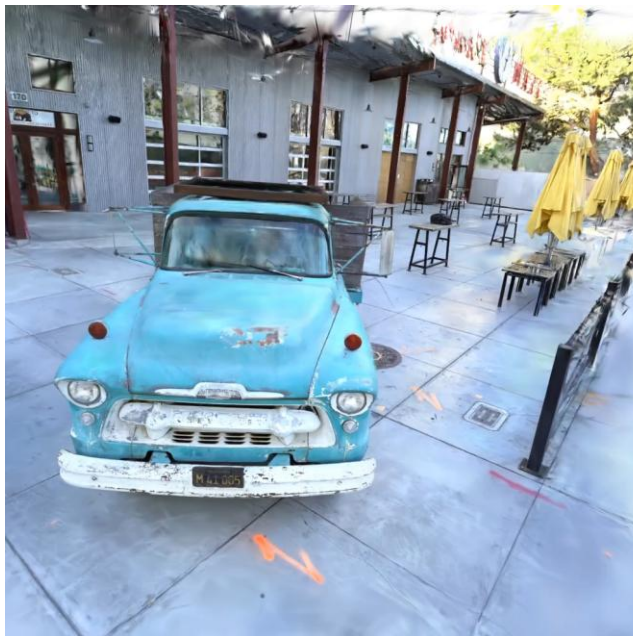
Ours



PSNR 24.83dB

Coding Results

3DGS



SOG



639.7 MB

22x reduction

28.8 MB

<https://survey.3dgs.zip/>



3DGS.zip: A survey on 3D Gaussian Splatting Compression Methods

Milena Bagdasarian¹, Paul Knoll¹, Yi-Hsin Li³, Florian Barthel^{1,2}, Anna Hilsmann¹, Peter Eisert^{1,2}, Wieland Morgenstern¹

¹ Fraunhofer Heinrich Hertz Institute, HHI

² Humboldt University of Berlin

³ Technical University Berlin

Star us on Github

arXiv

We present a work-in-progress survey on 3D Gaussian Splatting compression methods, focusing on their statistical performance across various benchmarks. This survey aims to facilitate comparability by summarising key statistics of different compression approaches in a tabulated format. The datasets evaluated include Tanks and Temples, Mip-NeRF 360, Deep Blending, and Synthetic NeRF. For each method, we report the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Learned Perceptual Image Patch Similarity (LPIPS), and the resultant size in megabytes (MB, 10⁶ bytes), as provided by the respective authors. This is an ongoing, open project, and we invite contributions from the research community as GitHub issues or pull requests.

Compression ☒ Compaction

Method	Rank	Tanks and Temples				Mip-NeRF 360				Deep Blending				Synthetic NeRF			
		PSNR	SSIM	LPIPS	Size [MB]	PSNR	SSIM	LPIPS	Size [MB]	PSNR	SSIM	LPIPS	Size [MB]	PSNR	SSIM	LPIPS	Size [MB]
HAC-lowrate	4.3	24.04	0.846	0.187	8.5	27.53	0.807	0.238	16.0	29.98	0.902	0.269	4.6	33.24	0.967	0.037	1.2
HAC-highrate	4.3	24.40	0.853	0.177	11.8	27.77	0.811	0.230	22.9	30.34	0.906	0.258	6.7	33.71	0.968	0.034	2.0
gsplat-1.00M	4.8	24.03	0.857	0.163	16.1	27.29	0.811	0.229	16.0								
IGS low	5.7	23.70	0.836	0.227	8.9	27.34	0.811	0.255	13.4	30.63	0.904	0.293	6.6	33.36	0.971	0.036	1.9
IGS high	6.4	24.05	0.849	0.211	13.1	27.62	0.820	0.245	27.0	32.33	0.924	0.253	8.1	34.18	0.975	0.032	2.9
Compact3D 32K	8.1	23.44	0.838	0.198	13.0	27.12	0.806	0.240	19.0	29.90	0.907	0.251	13.0				
CompGS	8.5	23.70	0.837	0.208	10.1	27.26	0.803	0.239	17.3	29.69	0.901	0.279	9.2				
Compact3D 16K	8.5	23.39	0.836	0.200	12.0	27.03	0.804	0.243	18.0	29.90	0.906	0.252	12.0				
RDO-Gaussian	9.1	23.34	0.835	0.195	12.0	27.05	0.802	0.239	23.5	29.63	0.902	0.252	18.0	33.12	0.967	0.035	2.3
Reduced3DGS	9.5	23.57	0.840	0.188	14.0	27.10	0.809	0.226	29.0	29.63	0.902	0.249	18.0				
SOG w/o SH	10.0	23.15	0.828	0.198	9.3	26.56	0.791	0.241	16.7	29.12	0.892	0.270	6.7	31.37	0.959	0.043	2.0

<https://survey.3dgs.zip/>



3DGS.zip: A survey on 3D Gaussian Splatting Compression Methods

Milena Bagdasarian¹, Paul Knoll¹, Yi-Hsin Li³, Florian Barthel^{1,2}, Anna Hilsmann¹, Peter Eisert^{1,2}, Wieland Morgenstern¹

¹ Fraunhofer Heinrich Hertz Institute, HHI

² Humboldt University of Berlin

³ Technical University Berlin

Star us on Github

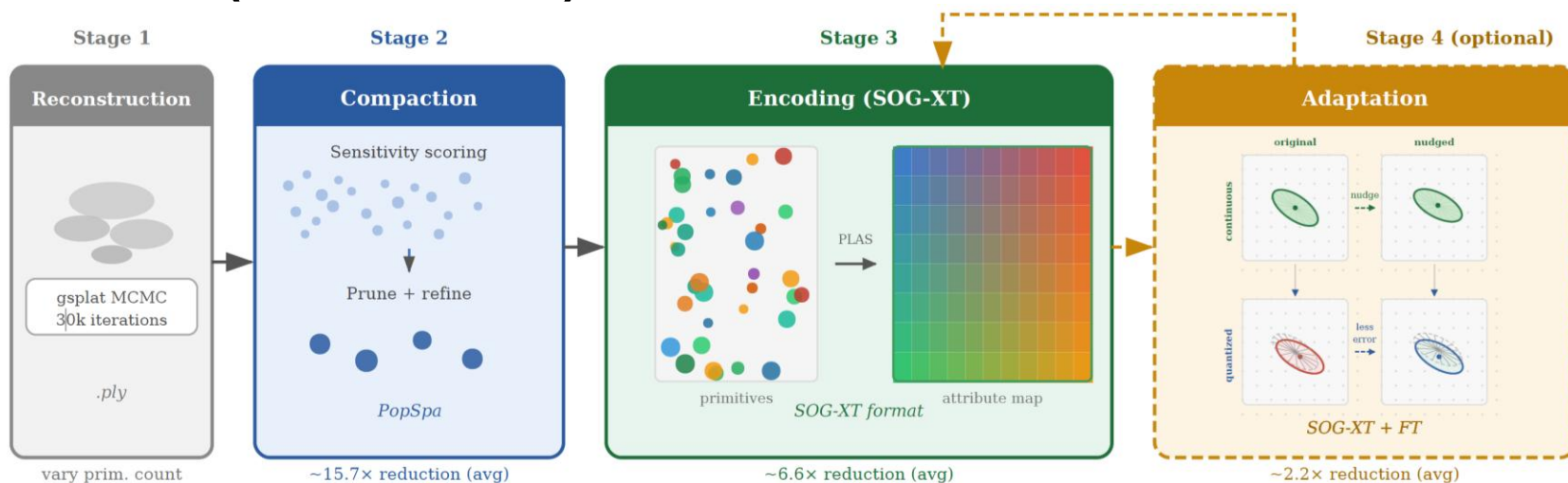
arXiv

We present a work-in-progress survey on 3D Gaussian Splatting compression methods, focusing on their statistical performance across various benchmarks. This survey aims to facilitate comparability by summarising key statistics of different compression approaches in a tabulated format. The datasets evaluated include Tanks and Temples, Mip-NeRF 360, Deep Blending, and Synthetic NeRF. For each method, we report the Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index (SSIM), Learned Perceptual Image Patch Similarity (LPIPS), and the resultant size in megabytes (MB, 10⁶ bytes), as provided by the respective authors. This is an ongoing, open project, and we invite contributions from the research community as GitHub issues or pull requests.

Compression ☒ Compaction

Method	Rank	Tanks and Temples				Mip-NeRF 360				Deep Blending				Synthetic NeRF			
		PSNR	SSIM	LPIPS	Size [MB]	PSNR	SSIM	LPIPS	Size [MB]	PSNR	SSIM	LPIPS	Size [MB]	PSNR	SSIM	LPIPS	Size [MB]
HAC-lowrate	4.3	24.04	0.846	0.187	8.5	27.53	0.807	0.238	16.0	29.98	0.902	0.269	4.6	33.24	0.967	0.037	1.2
HAC-highrate	4.3	24.40	0.853	0.177	11.8	27.77	0.811	0.230	22.9	30.34	0.906	0.258	6.7	33.71	0.968	0.034	2.0
gsplat-1.00M	4.8	24.03	0.857	0.163	16.1	27.29	0.811	0.229	16.0								
IGS low	5.7	23.70	0.836	0.227	8.9	27.34	0.811	0.255	13.4	30.63	0.904	0.293	6.6	33.36	0.971	0.036	1.9
IGS high	6.4	24.05	0.849	0.211	13.1	27.62	0.820	0.245	27.0	32.33	0.924	0.253	8.1	34.18	0.975	0.032	2.9
Compact3D 32K	8.1	23.44	0.838	0.198	13.0	27.12	0.806	0.240	19.0	29.90	0.907	0.251	13.0				
CompGS	8.5	23.70	0.837	0.208	10.1	27.26	0.803	0.239	17.3	29.69	0.901	0.279	9.2				
Compact3D 16K	8.5	23.39	0.836	0.200	12.0	27.03	0.804	0.243	18.0	29.90	0.906	0.252	12.0				
RDO-Gaussian	9.1	23.34	0.835	0.195	12.0	27.05	0.802	0.239	23.5	29.63	0.902	0.252	18.0	33.12	0.967	0.035	2.3
Reduced3DGS	9.5	23.57	0.840	0.188	14.0	27.10	0.809	0.226	29.0	29.63	0.902	0.249	18.0				
SOG w/o SH	10.0	23.15	0.828	0.198	9.3	26.56	0.791	0.241	16.7	29.12	0.892	0.270	6.7	31.37	0.959	0.043	2.0

KISS-GS (ECCV 2026)



- Compaction: combination GaussianSpa with GaussianPop
- Encoding: high dimensional sorting, SH via codebooks
- Adaptation: coding aware finetuning
- Total reduction

15.7 x reduction
6.6 x reduction
2.2 x reduction
228 x reduction

POPSpa Compaction

- GaussianPOP

- determine each Gaussian's contribution to the rendered views

- Constrained optimization with GaussianSPA:

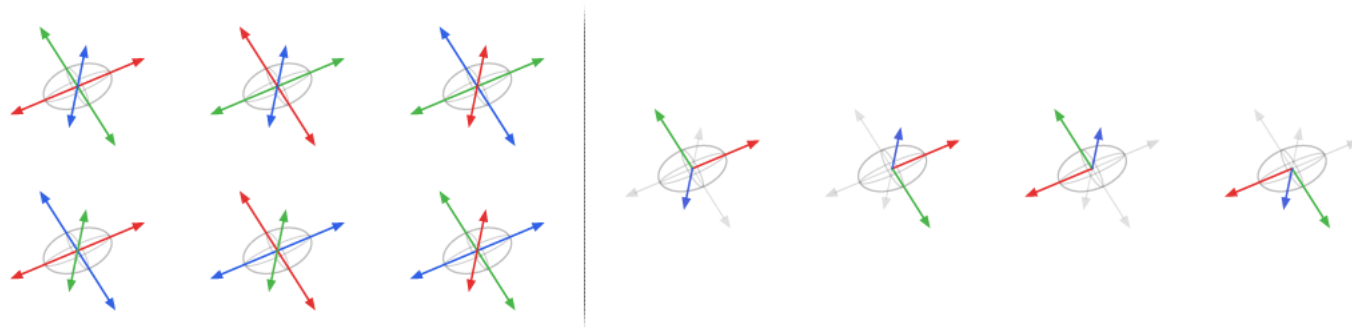
$$\min_{\mathbf{a}, \Theta} \mathcal{L}(\mathbf{a}, \Theta), \quad \text{s.t. } \|\mathbf{a}\|_0 \leq \kappa$$

- Efficient rank regularization (to avoid needle like Gaussians)

$$\text{erank}(\Sigma) = \exp\left(-\sum_{i=1}^3 p_i \cdot \log p_i\right)$$

- Largest gains (about 15x)

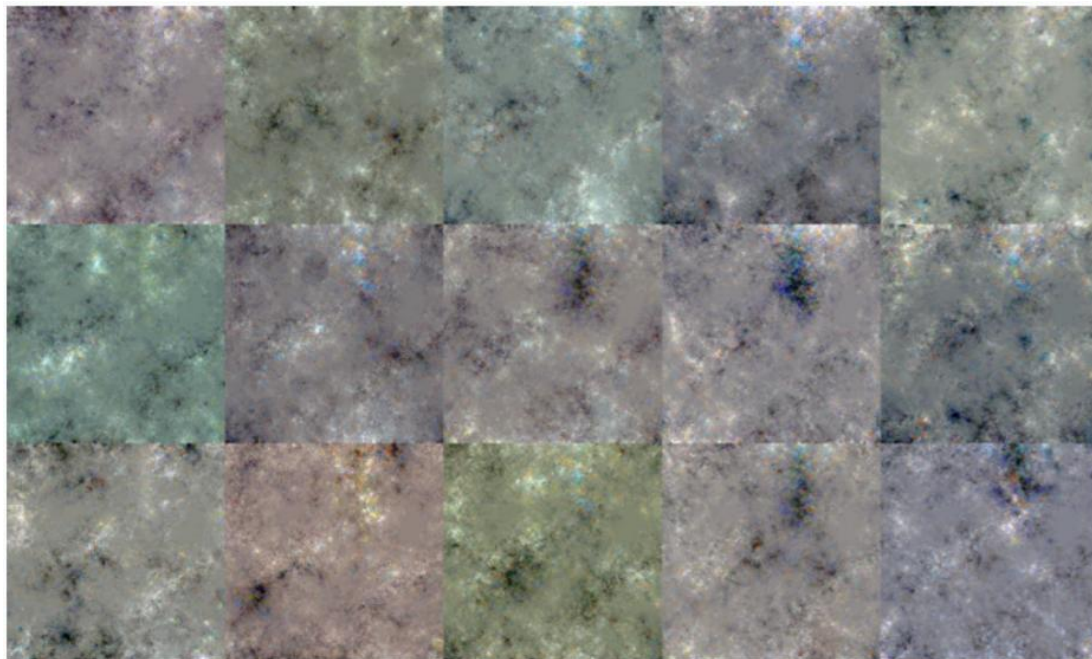
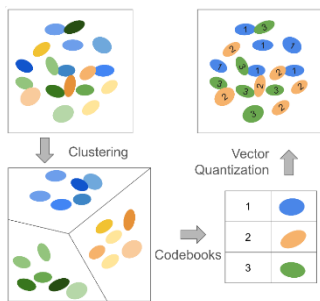
Ambiguities in Scale and Rotation



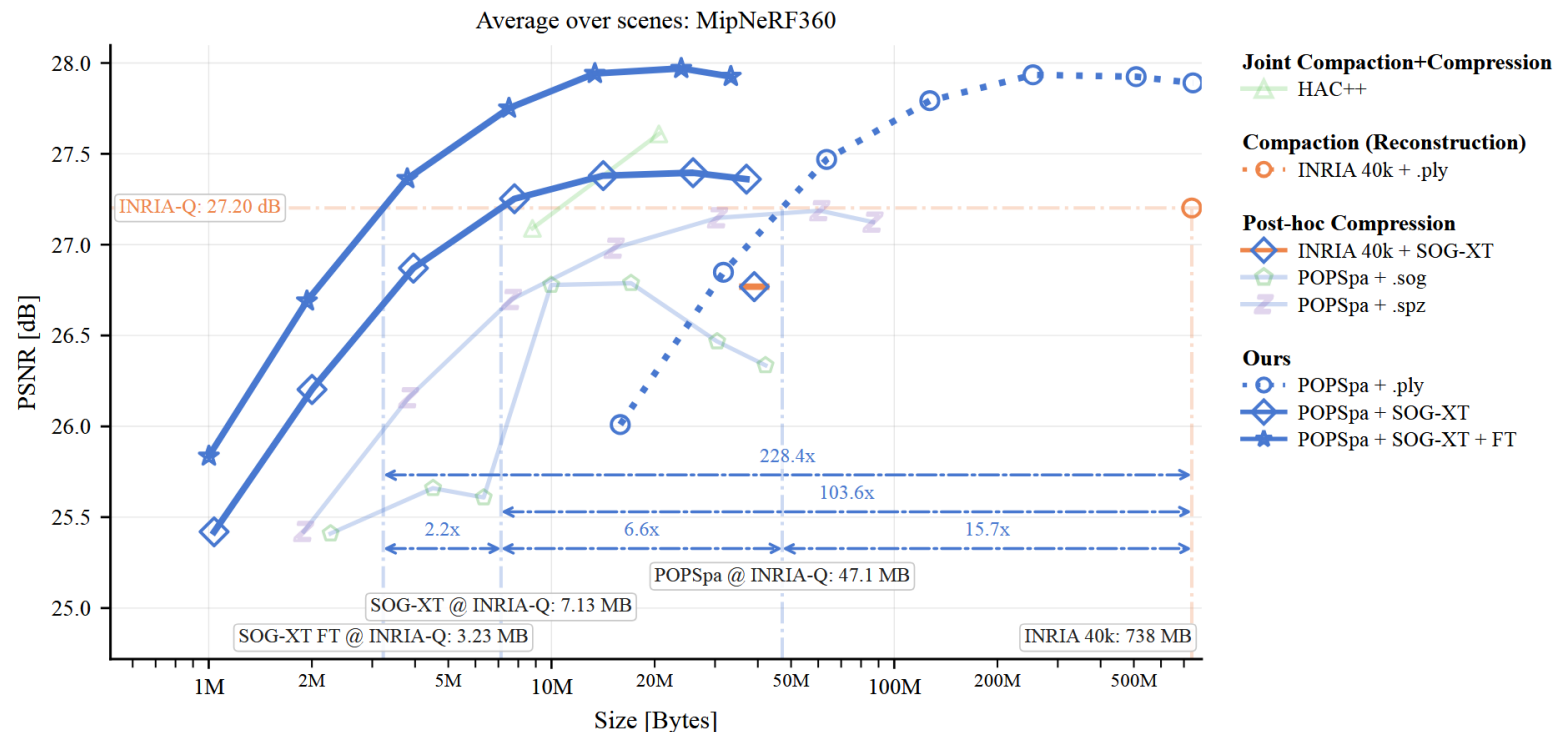
- Covariance is decomposed into $\Sigma = \mathbf{R}\mathbf{S}\mathbf{S}\mathbf{R}^T$
 - rotation (quaternion: 4 parameters) and
 - scale (3 parameters)
- 48 different parameter configurations with the same resulting Gaussian
- During sorting: assign the configuration with the highest smoothness

Spherical Harmonics: VQ with Sorted Codebooks

- Vector quantization of SH coefficients (15x3 per Gaussian)
- k-means clustering (64k centroids)
- PLAS sorting of codebook
 - codebook easy to encode
 - neighboring indices smoother



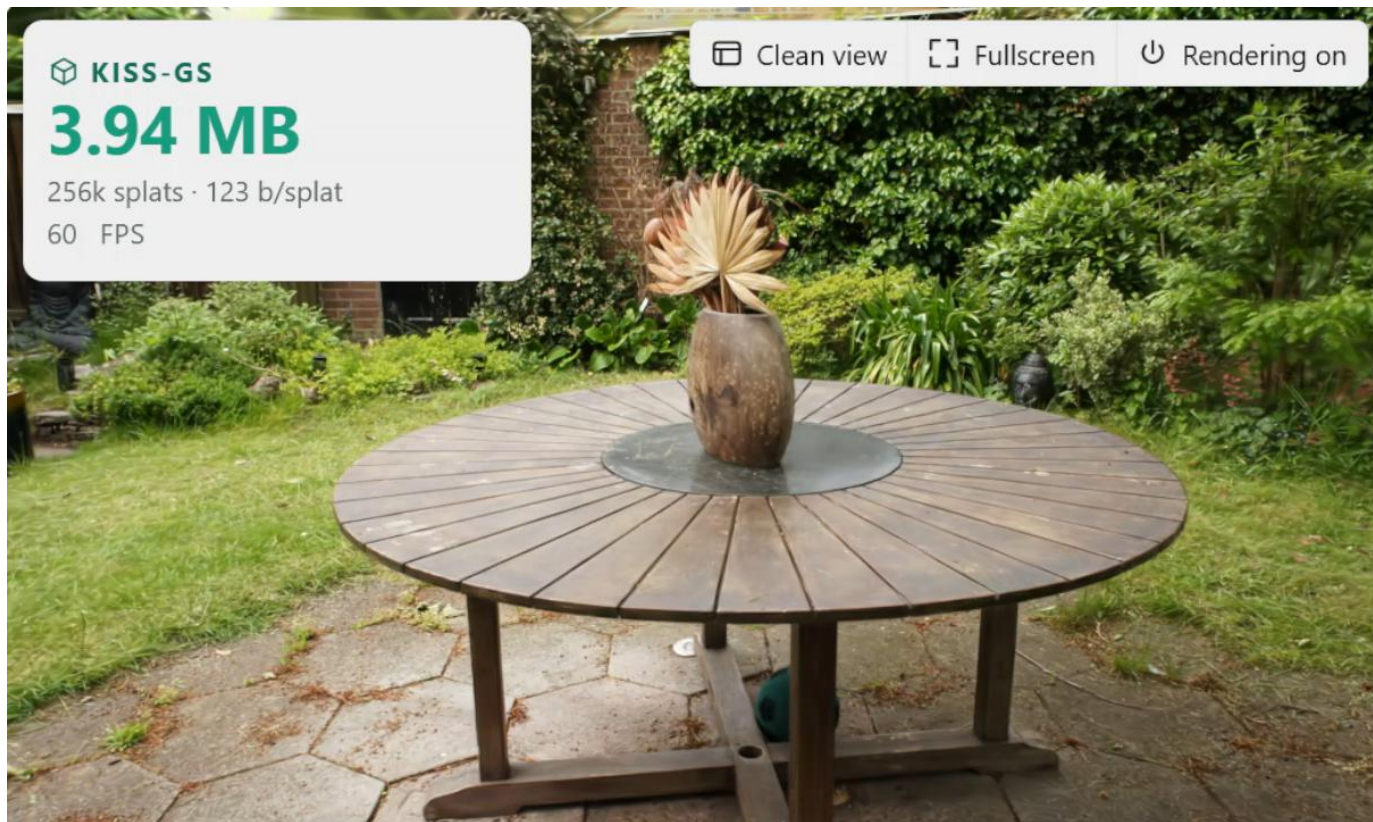
Results for MipNeRF360



Results and Evaluation

Method	Tanks and Temples			Mip-NeRF 360			Deep Blending			Synthetic NeRF		
	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS	PSNR	SSIM	LPIPS
CodecGS [21] ICCV '25 [†]	53*	-	-	72*	72*	-	76*	76*	-	n/a	n/a	n/a
ContextGS [42] NeurIPS '24 [†]	42*	42*	-	55*	55*	-	187*	187*	-	n/a	n/a	n/a
gsplat-1M [13] arXiv '24 [†]	60*	44	26	52	57	28	n/a	n/a	n/a	n/a	n/a	n/a
HAC [3] ECCV '24 [†]	49*	48	-	46*	46*	-	150*	129	-	47	-	-
HEMGS [22] arXiv '24 [†]	69*	69*	-	59*	59*	-	229*	229*	-	57*	-	-
HAC++ [5] TPAMI '25	77*	68	-	70	36	-	156	156	-	-	-	-
Ours (SOG-XT)	227	104	60	104	73	40	-	-	-	10	-	-
Ours+FT	319	131	72	228	109	64	85	98	-	21	14	14

Visual Result

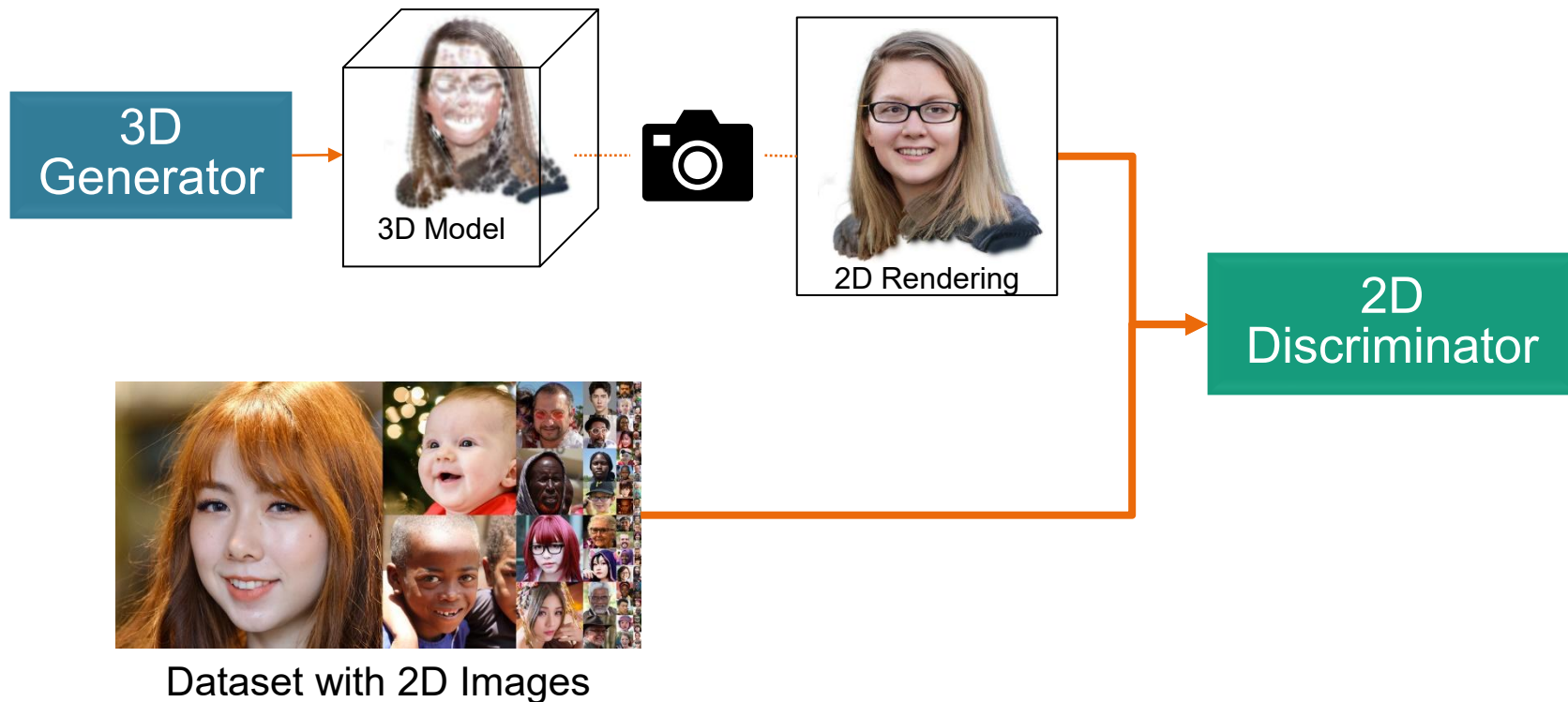


COSY: Compositional 3DGS Synthesis for Disentangled Human Head Editing [ECCV 2026]



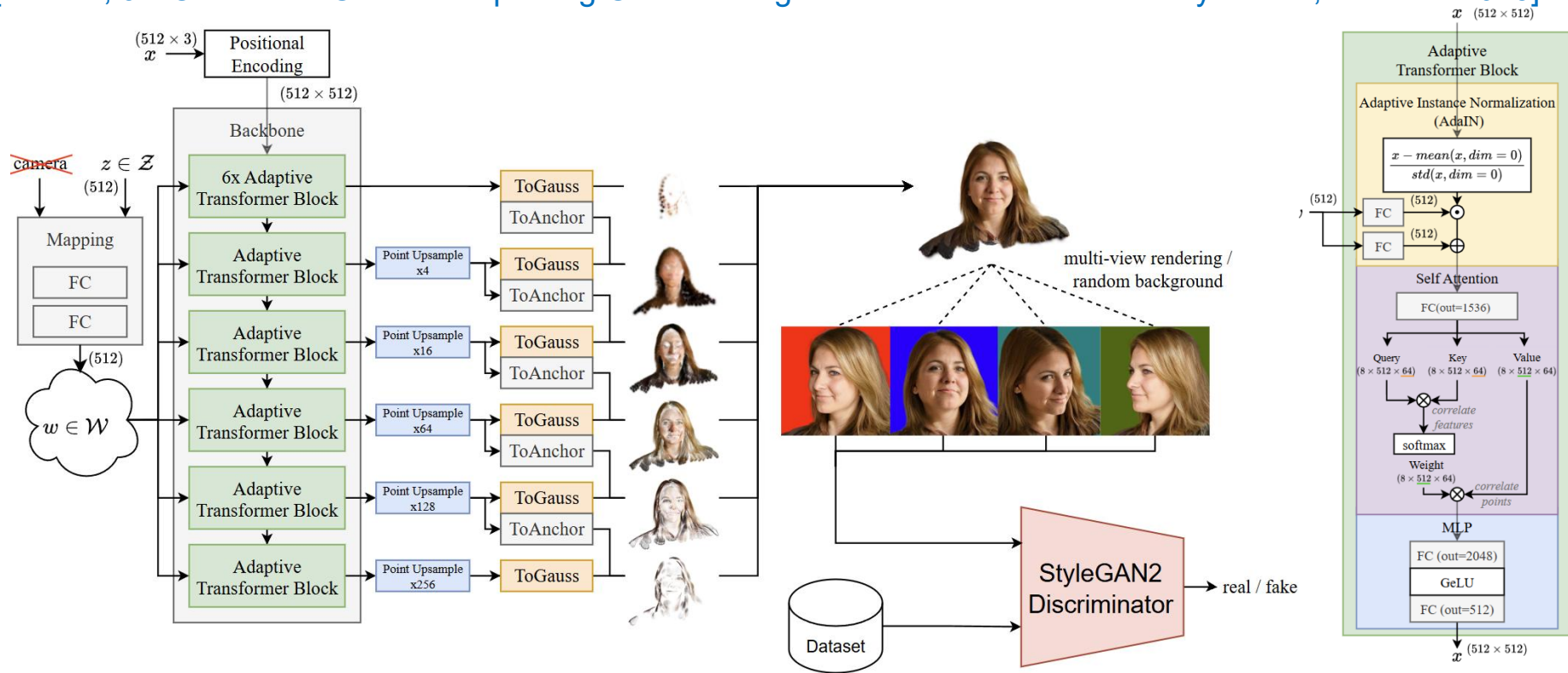
3D GANs for Head Model Synthesis

Learning from 2D images only



CGS-GAN

[Barthel, 3D Consistent Gaussian Splatting GANs for High Resolution Human Head Synthesis, NeurIPS 2025]

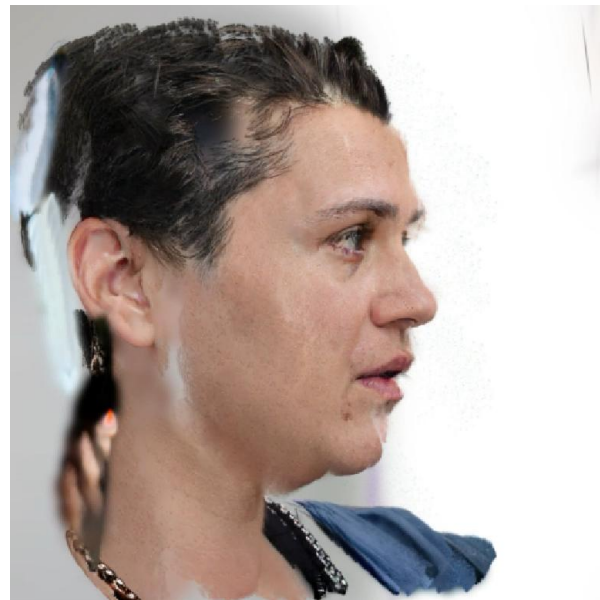


Current Problems with View Conditioning

Solved by removing View Conditioning and Multiview Regularization



Fixed Frontal Conditioning
(only frontal views look good)



Standard Viewpoint Conditioning
(inconsistent over viewing direction)

Comparison of CSGAN with other Approaches



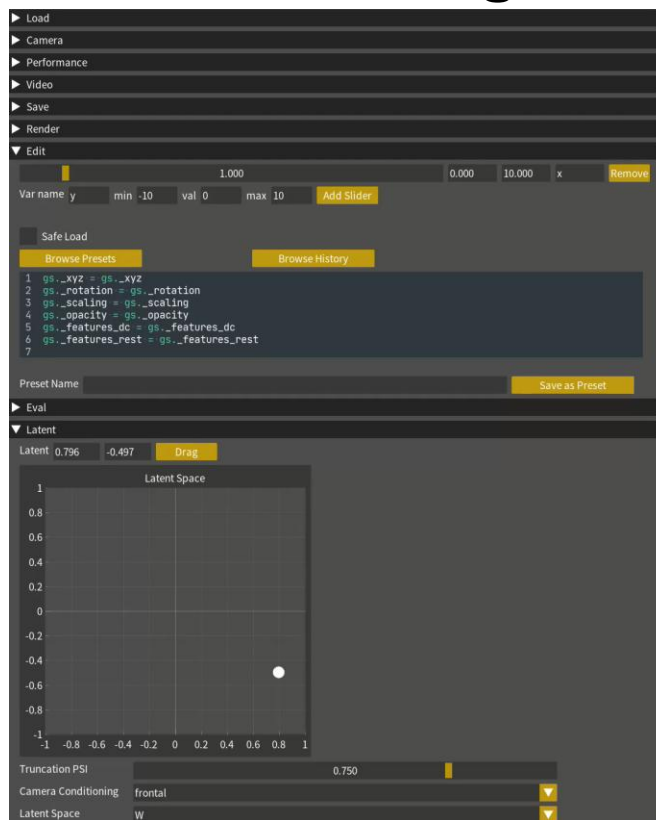
GGHead

GSGAN

ours

ours
fixed data

Interactive Editing



[NeurIPS 2025]

Fitting to Images through GAN Inversion

▼ Load

Load pkl

► Camera

► Performance

► Save

► Render

► Edit

► Eval

► Latent

▼ Inversion

1. Load Images

Open Images from Files

► Webcam

2. Image Preprocessing

Preprocess

3. Run Latent Inversion

ID Similarity	0.10000	0.10000	
MSE	0.01000	0.01000	
LPIPS	1.00000	1.00000	
Learning Rate	0.00050	0.00050	
Batch Size	1	1	

Start Inversion

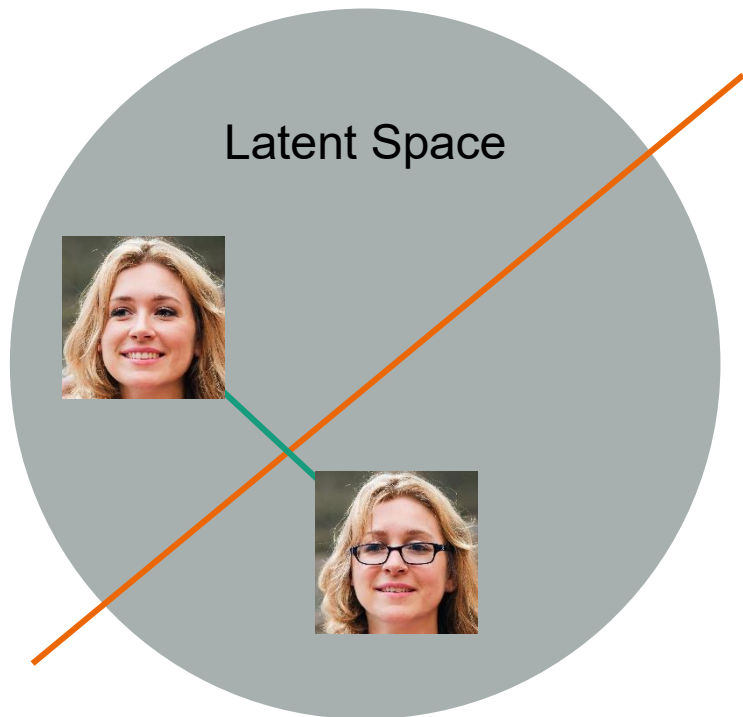
4. Run Generator Tuning

ID Similarity	0.10000	0.10000	
MSE	0.01000	0.01000	
LPIPS	1.00000	1.00000	
Learning Rate	0.00050	0.00050	
Batch Size	1	1	

Start Tuning



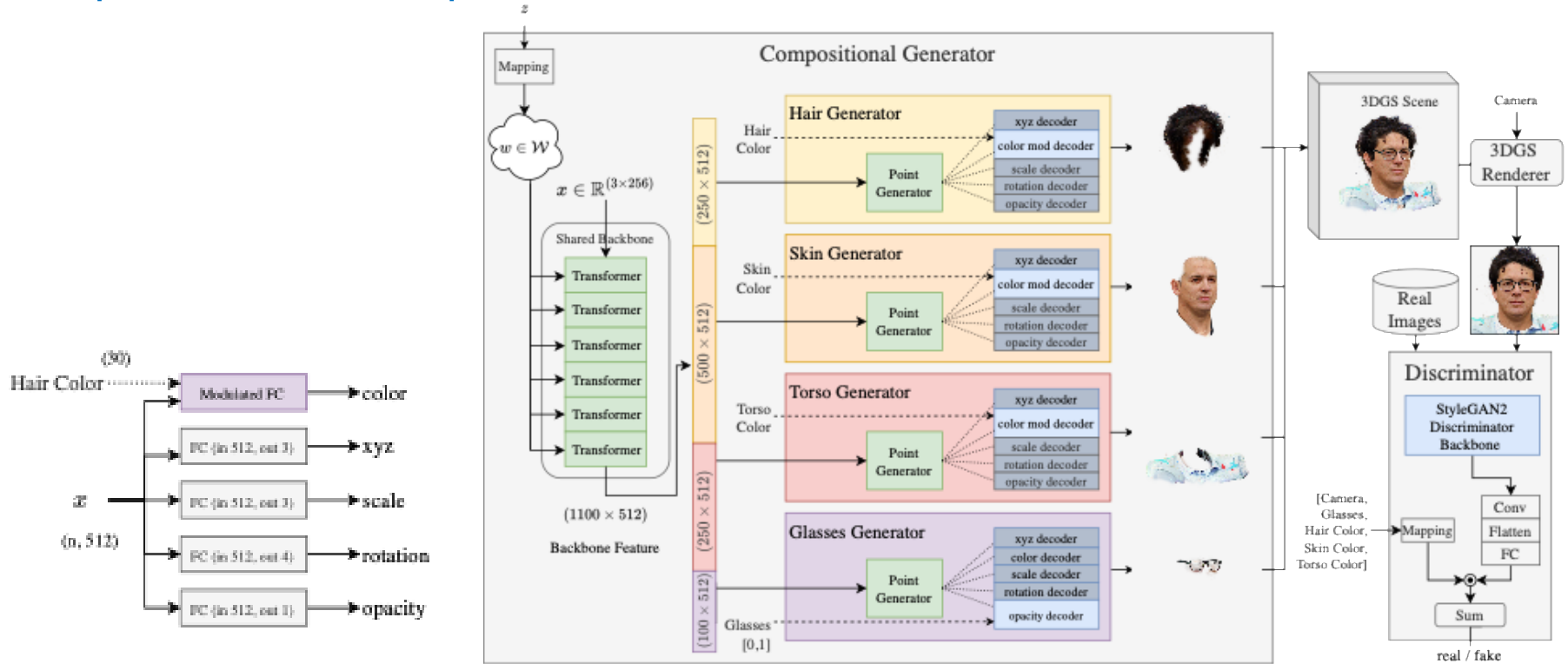
Problem: Entangled GAN Space



[StyleFlow]

COSY [Barthel et al. ECCV 2026]

Independent Latent Spaces for different Attributes

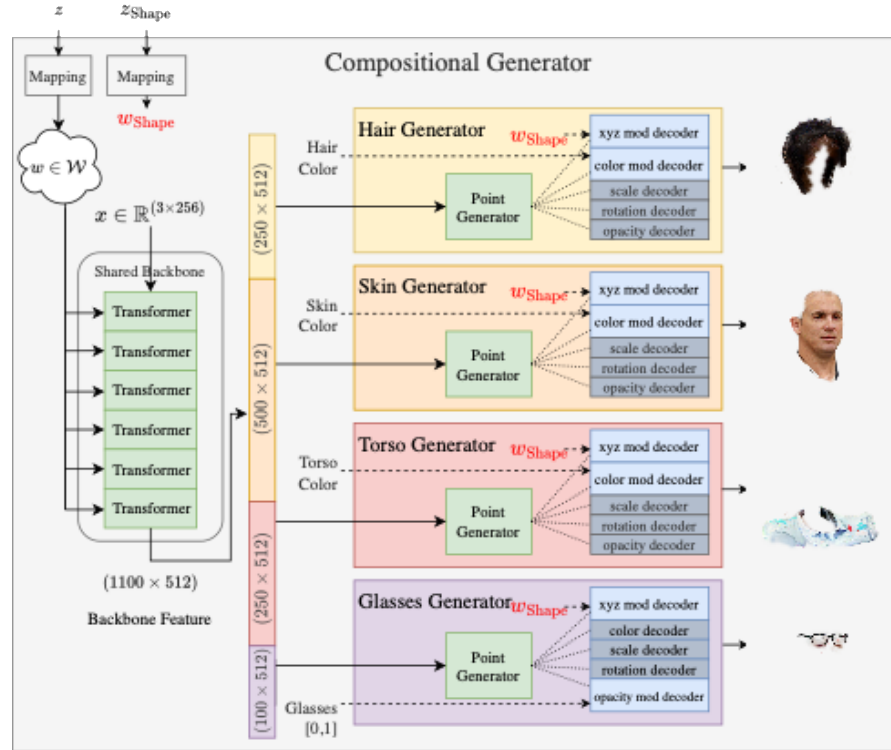


Shape Context

Ensures that different parts fit together



- Latent shape vector
- Only affects small decoder of positions
- Cannot change colors etc.



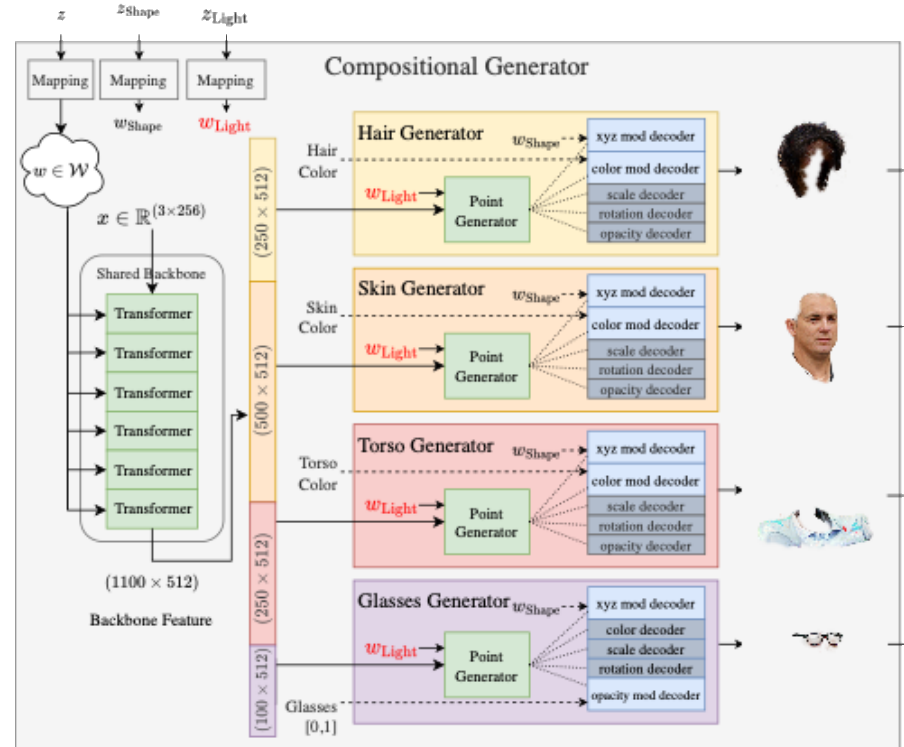
Light Context

Ensures that lighting is consistent

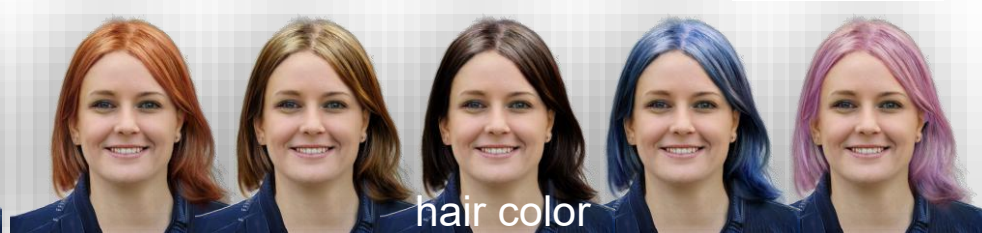
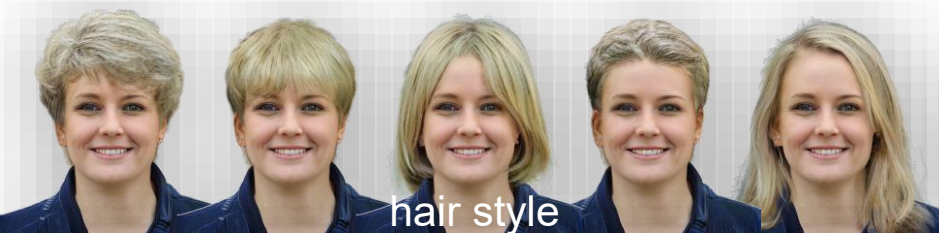


- Latent light vector
- Input to point generator to generate complex shadows
- Regularized:

$$\text{Reg} = \text{LPIPS}(\text{w_light1}, \text{w_light2})$$



Disentangled Editing of 3D Models





Peter Elert – Efficient Coding and Synthesis of Gaussian Splatting Scenes

Future Work



other object classes



multiple object classes

Conclusion

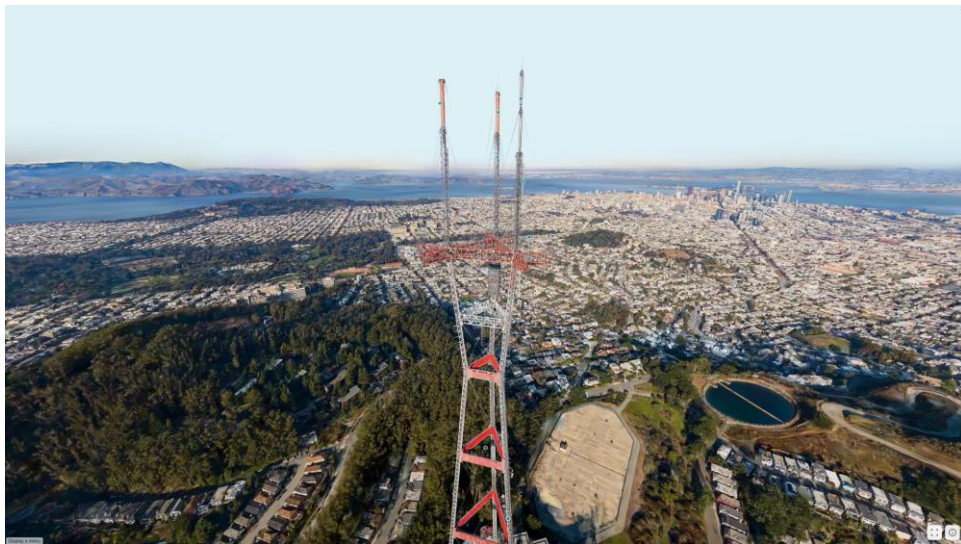
- Gaussian Splatting is an exciting representation but huge in size
 - High dimensional sorting enables efficient compression
 - Size reduction can be reached via compaction, encoding of adaptation (finetuning)
 - Sorting can also improve vector quantization
- Gaussian Splats can also be synthesized via GANs
 - No 3D datasets required
 - Fast and interactive editing
 - Disentangling of latent space into sub-groups for easier modification
 - No mask or object supervision

Thanks to all the Collaborators

- Milena Bagdasarian
- Florian Barthel
- Margherita Corona
- Paul Hinzer
- Anna Hilsmann
- Wieland Morgenstern
- Paul Schlack



Questions?



Contact:

Peter Eisert
peter.eisert@hu-berlin.de
+49 30 2093 41180



Federal Ministry
for Economic Affairs
and Energy

References

Literature I

- [\[2DGS\]](#) Huang, Yu, Chen, Geiger, Gao, 2D Gaussian Splatting for Geometrically Accurate Radiance Fields, Siggraph 2024.
- [\[3DGS\]](#) Kerbl, Kopanas, Leimkühler, Drettakis, 3d gaussian splatting for real-time radiance field rendering. ACM Trans. on Graphics 42, 2023
- [\[3DGS.zip\]](#) M. Bagdasarian, P. Knoll, Y. Li, F. Barthel, A. Hilsmann, P. Eisert, W. Morgenstern, 3DGS.zip: A survey on 3D Gaussian Splatting Compression Methods, Proc. Eurographics, London, UK, May 2025. doi: 10.1111/cgf.70078.
- [\[4DGS\]](#) Wu, Yi, Fang, Xie, Zhang, Wei, Liu, Tian, Wang, Xinggang, 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering, CVPR 2024.
- [\[AtomGS\]](#) Liu, Xu, Hu, Chen, Feng, AtomGS: Atomizing gaussian splatting for high-fidelity radiance field. arXiv 2024.
- [\[CGSGAN\]](#) Barthel, Morgenstern, Hinzer, Hilsmann, Eisert, GS-GAN: 3D Consistent Gaussian Splatting GANs for High Resolution Human Head Synthesis, NeurIPS 2025.
- [\[CompactGS\]](#) Lee, Rho, Sun, Ko, Park, Compact 3d gaussian representation for radiance field. CVPR 2024.
- [\[CodecGS\]](#) Lee, Shu, Sanchez de la Fuente, Schierl, Hellge, Compression of 3D Gaussian Splatting with Optimized Feature Planes and Standard Video Codecs. arXiv 2025.

Literature II

- [\[CompactGS\]](#) Lee, Rho, Sun, Ko, Park, Compact 3d gaussian representation for radiance field. CVPR 2024.
- [\[Compressed3D\]](#) Niedermayr, Stumpfegger, Westermann, Compressed 3D Gaussian Splatting for Accelerated Novel View Synthesis, CVPR 2024.
- [\[CompGS\]](#) Liu, Wu, Zhang, Wang, Li, Kwong, CompGS: Efficient 3d scene representation via compressed gaussian splatting. ACMM 2024.
- [\[ColorCuedGS\]](#) Kim, Lee, Lee, Color-cued efficient densification method for 3d gaussian splatting, CVPR 2024.
- [\[DEG\]](#) Hezel, Barthel, Schilling, Schall, Jung,. Dynamic Exploration Graph: A Novel Approach for Efficient Nearest Neighbor Search in Evolving Multimedia Datasets. MMM 2025.
- [\[Eagles\]](#) Girish, Gupta, Shrivastava, Eagles: Efficient accelerated 3d gaussians with lightweight encodings. ECCV 2024.
- [\[FLAS\]](#) Barthel, Hezel, Jung, Schall. Improved evaluation and generation of grid layouts using distance preservation quality and linear assignment sorting. Computer Graphics Forum, 2023.
- [\[EWASplatting\]](#) Zwicker, Pfister, Van Baar, Gross, EWA Splatting, IEEE Trans. TVCG 2002
- [\[FreGS\]](#) Zhang, Zhan, Xu, Lu, Xing, FreGS: 3d gaussian splatting with progressive frequency regularization, CVPR 2024.
- [\[GaussianPro\]](#) Chen, Long, Yang, Yao, Yin, Ma, Wang, Chen, GaussianPro: 3d gaussian splatting with progressive propagation, ICML 2024.

Literature III

- [\[GradSort\]](#) Barthel, Barthel, Eisert, Hezel, Schall. Creating sorted grid layouts with gradient-based optimization. ICMR '24,
- [\[GradSortN\]](#) Barthel, Barthel, Eisert, Permutation Learning with Only N Parameters: From SoftSort to Self-Organizing Guayanas, EUSIPCO, 2025.
- [\[HAC\]](#) Chen, Wu, Lin Harandi, Cai, HAC :Hash-grid assisted context for 3d gaussian splatting compression, ECCV 2024.
- [\[HAC++\]](#) Chen, Wu, Lin, Harandi, Cai, HAC++: Towards 100X Compression of 3D Gaussian Splatting, arxiv 2025.
- [\[HierarchicalGS\]](#) B. Kerbl, A. Meulemann, G. Kopanas, M. Wimmer, A. Lanvin, G. Drettakis, A Hierarchical 3D Gaussian Representation for Real-Time Rendering of Very Large Datasets, Siggraph 2024.
- [\[ImprovedADC\]](#) Grubert, Barthel, Hilsmann, Eisert, Improving adaptive density control for 3D gaussian splatting. VISAPP 2025.
- [\[IsoMatch\]](#) Fried, DiVerdi, Halber, Sizikova, Finkelstein,,: Isomatch: Creating informative grid layouts, Computer Graphics Forum 2015.
- [\[InstantNGP\]](#) Müller, Evans, Schied, Keller, Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. on Graphics 2022.

Literature IV

- **[LightGaussian]** Fan, Wang, Wen, Zhu, Xu, Wang, LightGaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps. Advances in Neural Information Processing Systems 37 (2024).
- **[MakingGSSmaller]** Aras Pranckevičius, Making Gaussian Splats smaller, <https://aras-p.info/blog/2023/09/13/Making-Gaussian-Splats-smaller/>, 2023
- **[MCDC]** Kheradmand, Rebain, Sharma, Sun, Tseng, Isack, Kar, Tagliasacchi, Yi: 3d gaussian splatting as markov chain monte carlo. NeurIPS 2024.
- **[MiniSplat]** Fang, Wang, Mini-splatting: Representing scenes with a constrained number of gaussians. ECCV 2024.
- **[MipNeRF360]** Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, Peter Hedman, Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields. CVPR 2022.
- **[MVGSPatting]** Li, Yao, Chum, Garcia-Fernandez, Yue, Lim, Zhu, MVG-Splatting: Multi-view guided gaussian splatting with adaptive quantile-based geometric consistency densification. arxiv 2024
- **[OctreeGS]** Ren, Jiang, Li, Yu, Xu, Ni, Dai, Octree-GS: Towards consistent real-time rendering with lod-structured 3d gaussians. arXiv 2024.
- **[PerspectiveSplat]** Hahlbohm, Friederichs, Weyrich, Franke, Kappel, Castillo, Stamminger, Eisemann, Magnor, Efficient Perspective-Correct 3D Gaussian Splatting Using Hybrid Transparency, Eurographics 2025

Literature V

- [\[PixelGS\]](#) Zhang, Hu, Lao, He, Zhao, Pixel-GS: Density control with pixel-aware gradient for 3d gaussian splatting, ECCV 2024.
- [\[R3GW\]](#) Corona, Morgenstern, Eisert, Hilsmann, R3GW: Relightable 3D Gaussians for Outdoor Scenes in the Wild, VISAPP 2026.
- [\[RevisingDens\]](#) Rota Bulo, Porzi, Kotschieder, Revising densification in gaussian splatting. ECCV 2024.
- [\[ScaffoldGS\]](#) Lu, Yu, Xu, Xiangli, Wang, Lin, Dai, Scaffold-GS: Structured 3d gaussians for view-adaptive rendering. CVPR 2024.
- [\[SoftSort\]](#) Prillo, Eisenschlos., Softsort: A continuous relaxation for the argsort operator. ICML 2020.
- [\[SOG\]](#) Morgenstern, Barthel, Hilsmann, Eisert, Compact 3d scene representation via self-organizing gaussian grids., ECCV 2024.
- [\[SOM\]](#) Kohonen. Essentials of the self-organizing map. Neural Networks, 2013.
- [\[SSM\]](#) Strong, Gong. Self-sorting map: An efficient algorithm for presenting multimedia data in structured layouts. IEEE Trans. Multim., 2014.

Literature VI

- [\[SuGaR\]](#) Guedon, Lepetit, SuGaR: Surface-Aligned Gaussian Splatting for Efficient 3D Mesh Reconstruction and High-Quality Mesh Rendering, CVPR 2024.
- [\[TamingGS\]](#) Mallick, Goel, Kerbl, Vincente Carrasco, Steinberger, De la Torre, Taming 3DGS: High-quality radiance fields with limited resources. SIGGRAPH Asia 2024.
- [\[tSNE\]](#) van der Maaten, Hinton. Visualizing data using t-SNE. Journal of Machine Learning Research, 2008.