EAGLES: Efficient Accelerated 3D Gaussians with Lightweight EncodingS

Sharath Girish, Kamal Gupta, Abhinav Shrivastava

University of Maryland, College Park

ECCV 2024

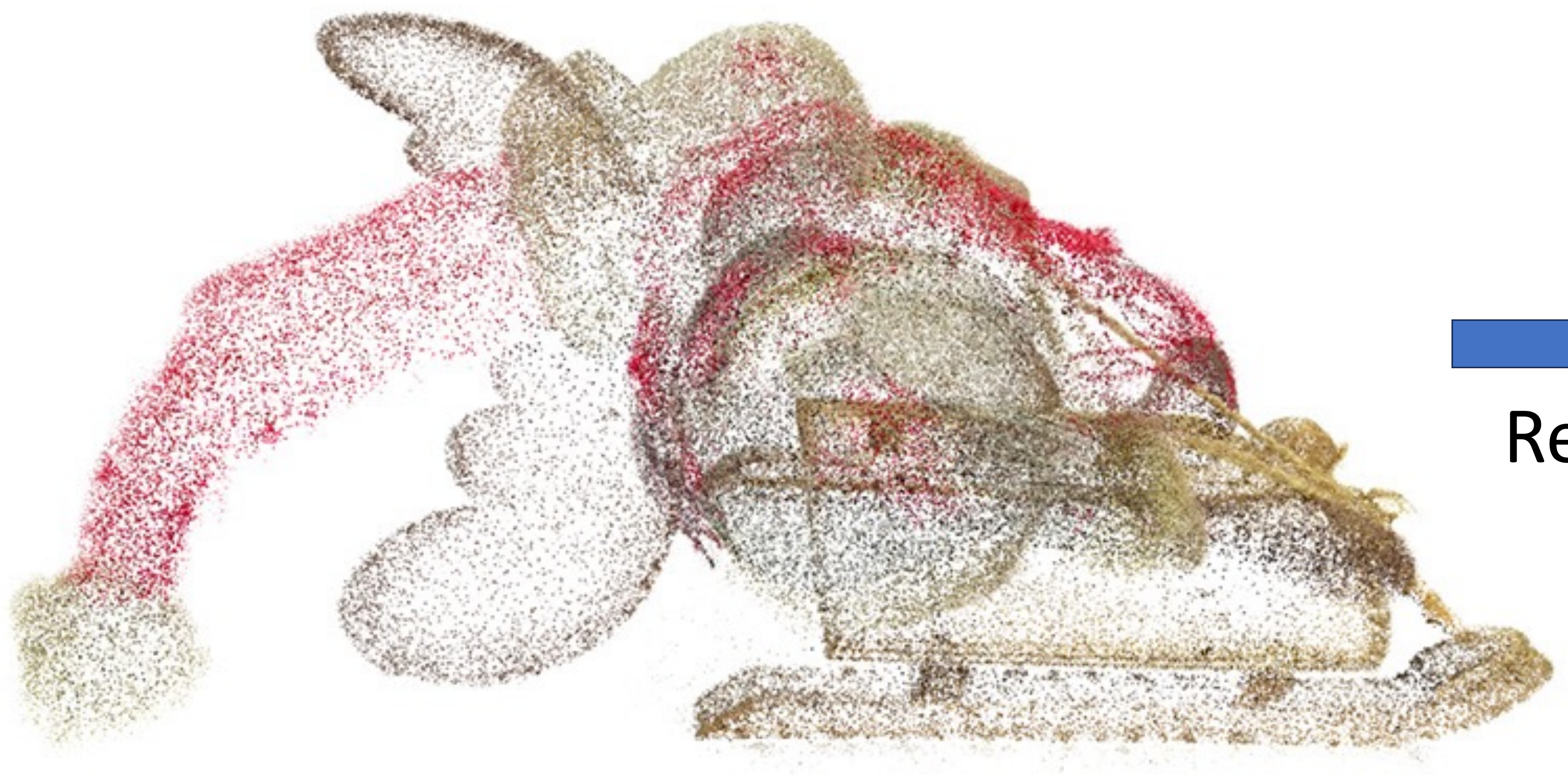
Novel view synthesis

Sparse view input of scene -> Novel view rendering

Sparse input views

Novel view reconstruction

3D Gaussian Splatting (3D-GS)



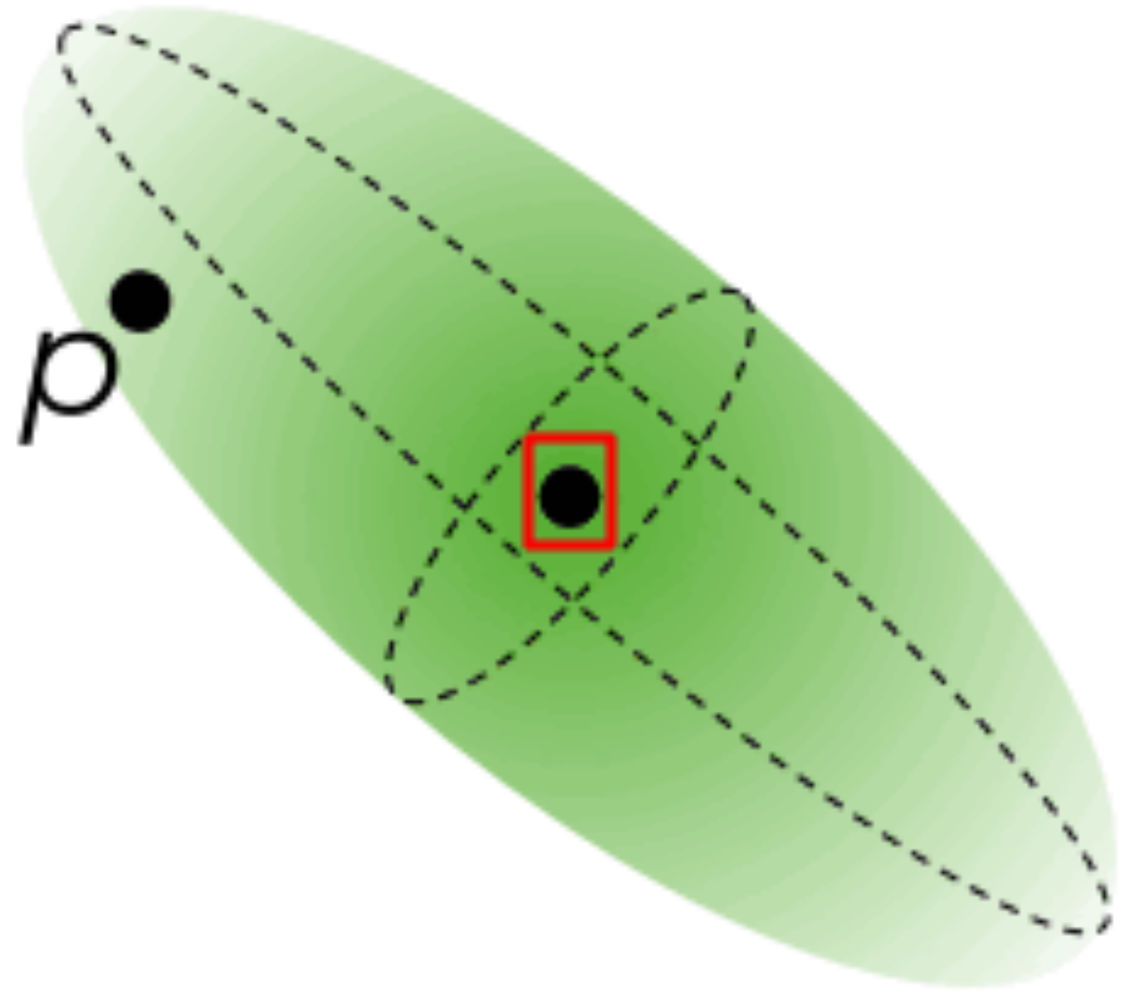
3D Gaussian point clouds

Novel view image synthesis

Kerbl, Bernhard, et al. "3D Gaussian Splatting for Real-Time Radiance Field Rendering." *ACM Trans. Graph.* 42.4 (2023): 139-1.

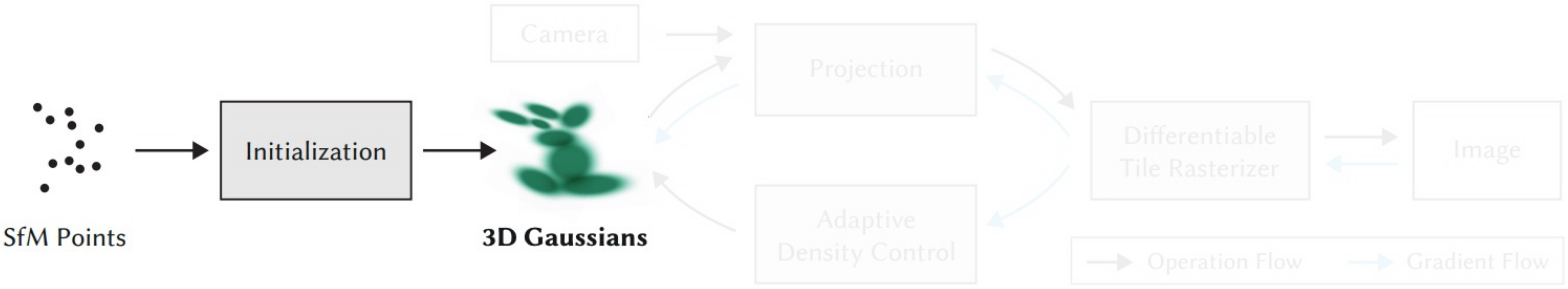
3D Gaussian attributes

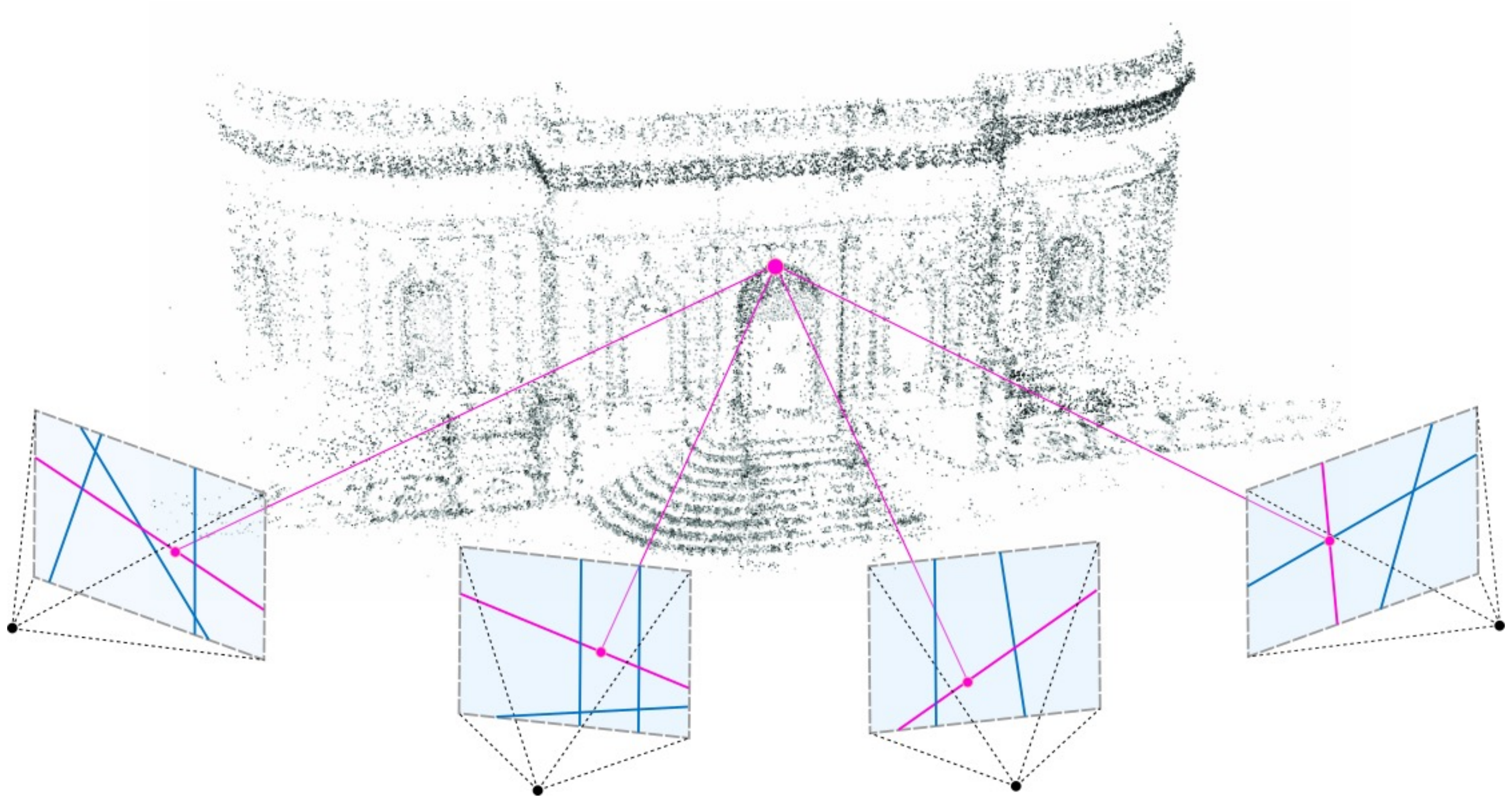
$$f_i(p) = \sigma(\alpha_i) \exp(-\frac{1}{2}(p - \boxed{\mu_i})\Sigma_i^{-1}(p - \boxed{\mu_i}))$$



$\boldsymbol{\mu}_i \in \mathbb{R}^3$	$\boldsymbol{q}_i \in \mathbb{R}^4$	$\boldsymbol{s}_i \in \mathbb{R}^3$	$\boldsymbol{\alpha}_i \in \mathbb{R}$	$\boldsymbol{c}_i \in \mathbb{R}^{3d^2}$
Position	Rotation	Scaling	Opacity	Color SH

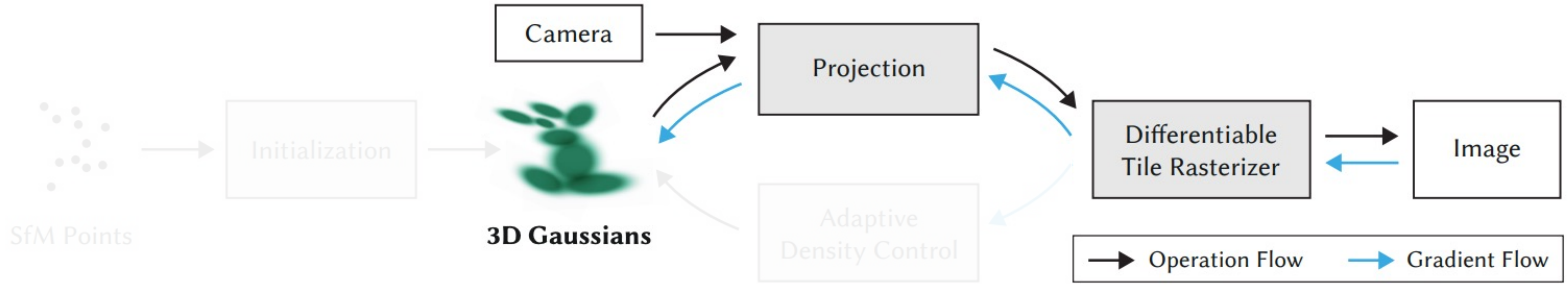
Colmap initialization

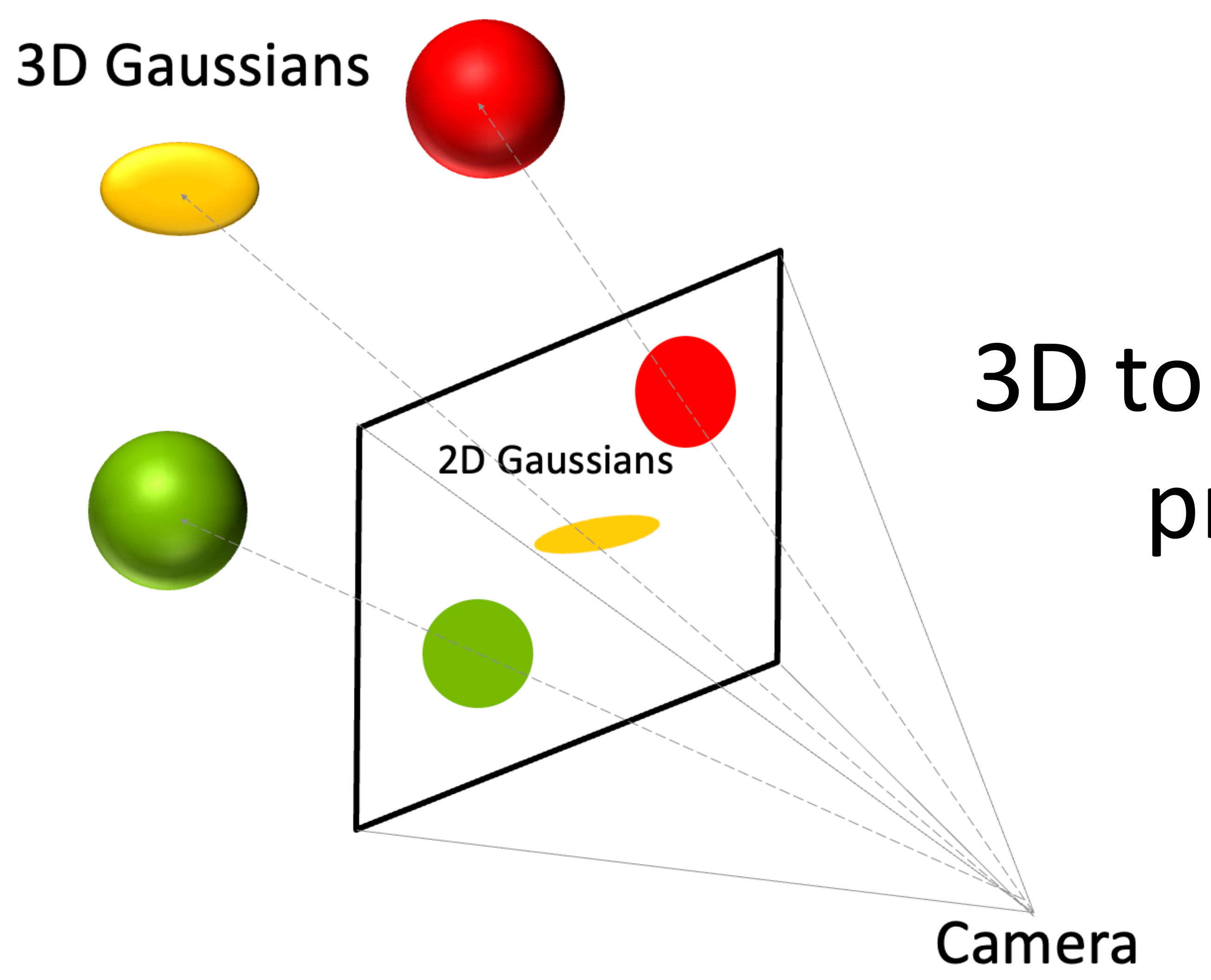




Colmap SFM initialization
of 3D Gaussian point clouds

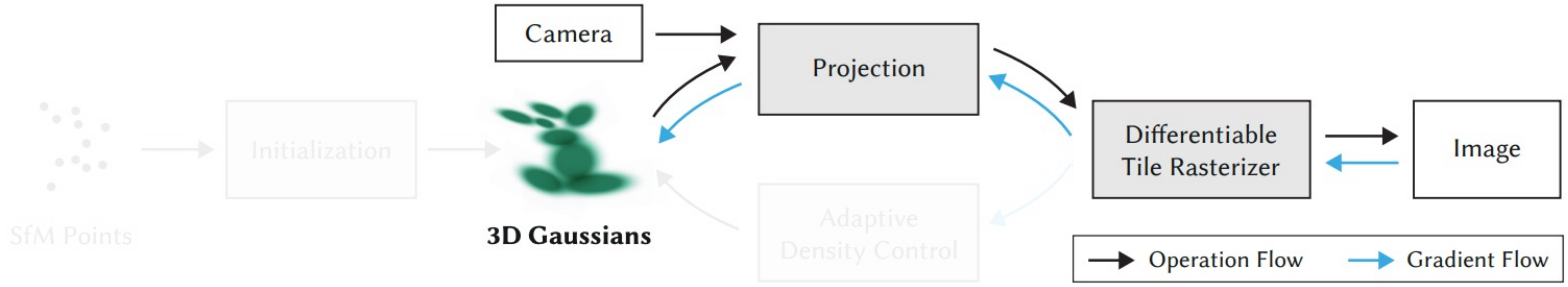
Image rasterization

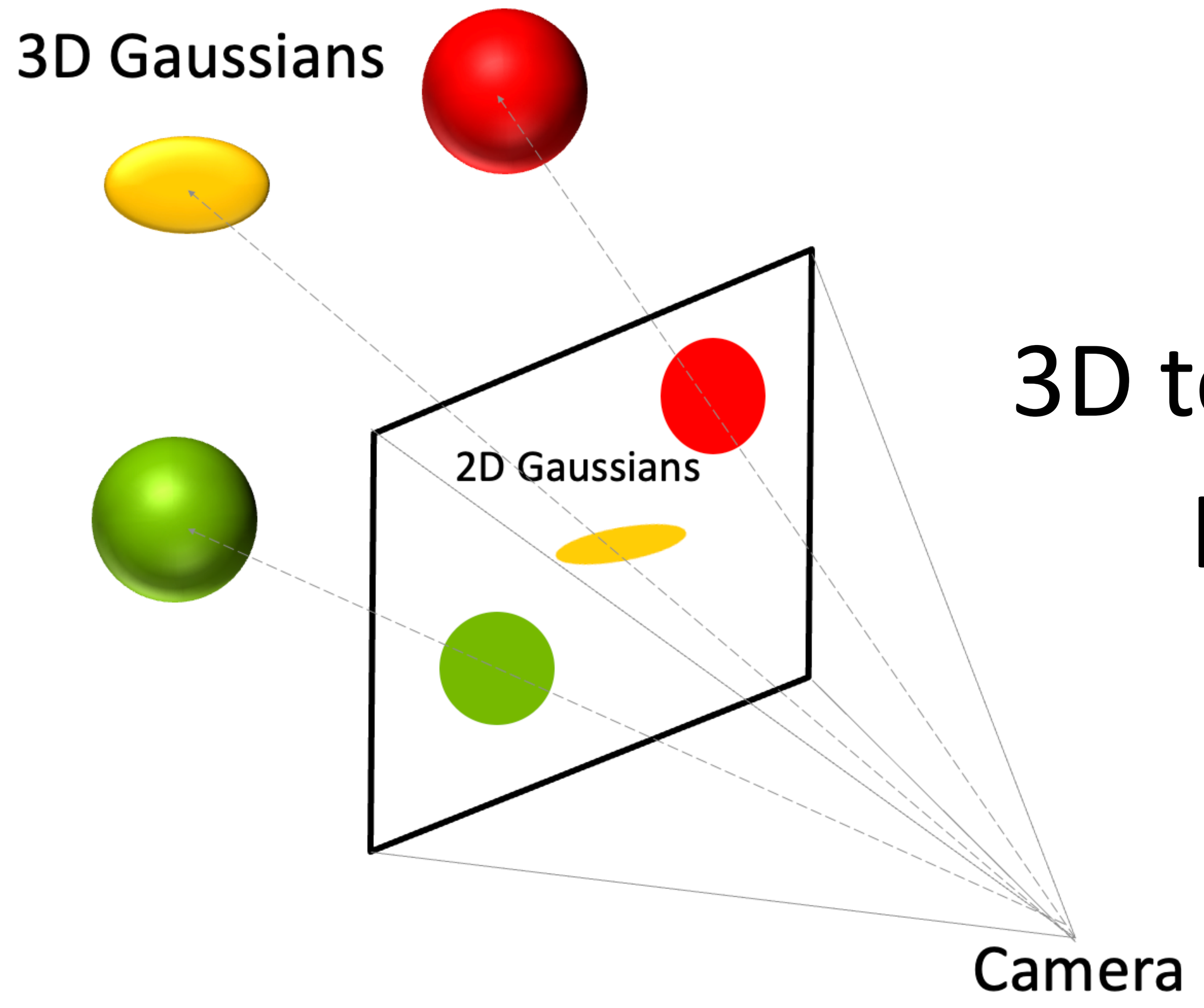




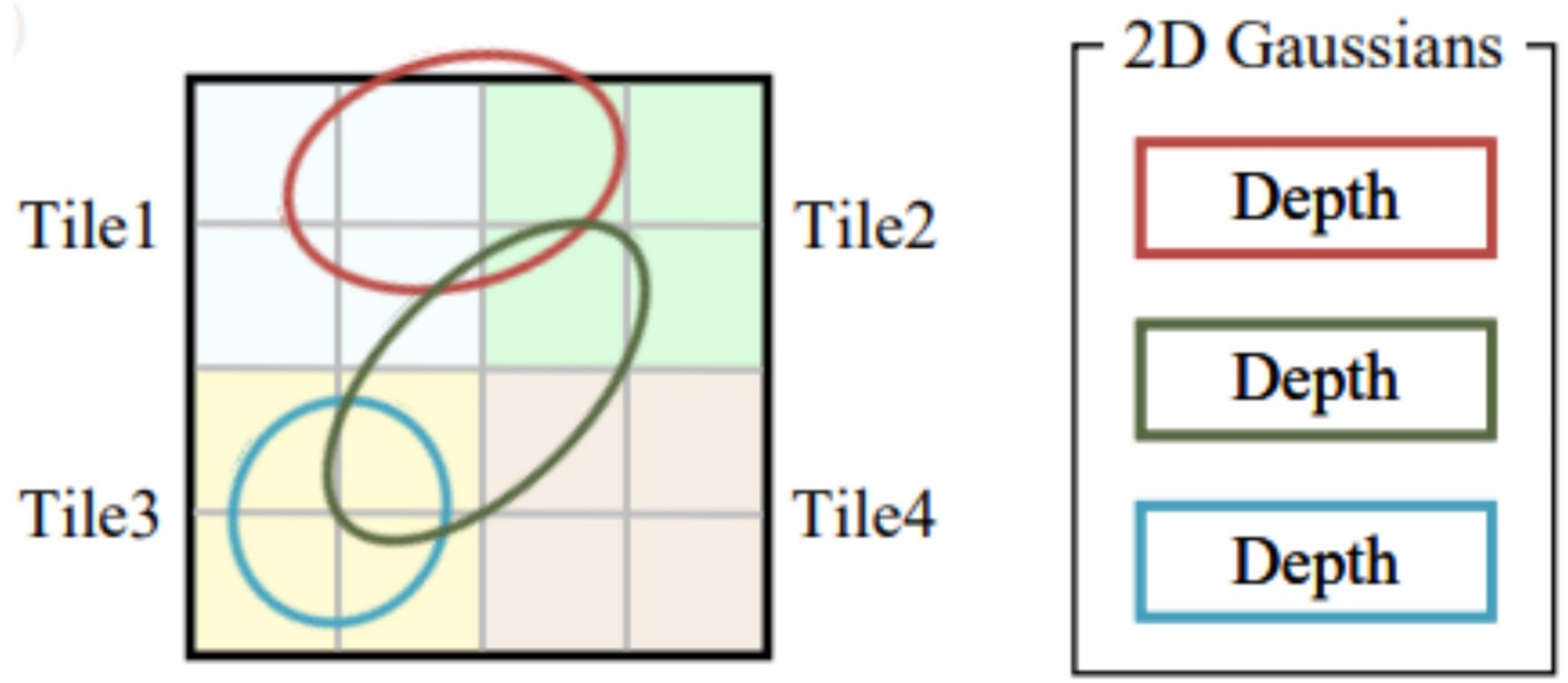
3D to 2D Gaussian projection

Image rasterization





3D to 2D Gaussian projection

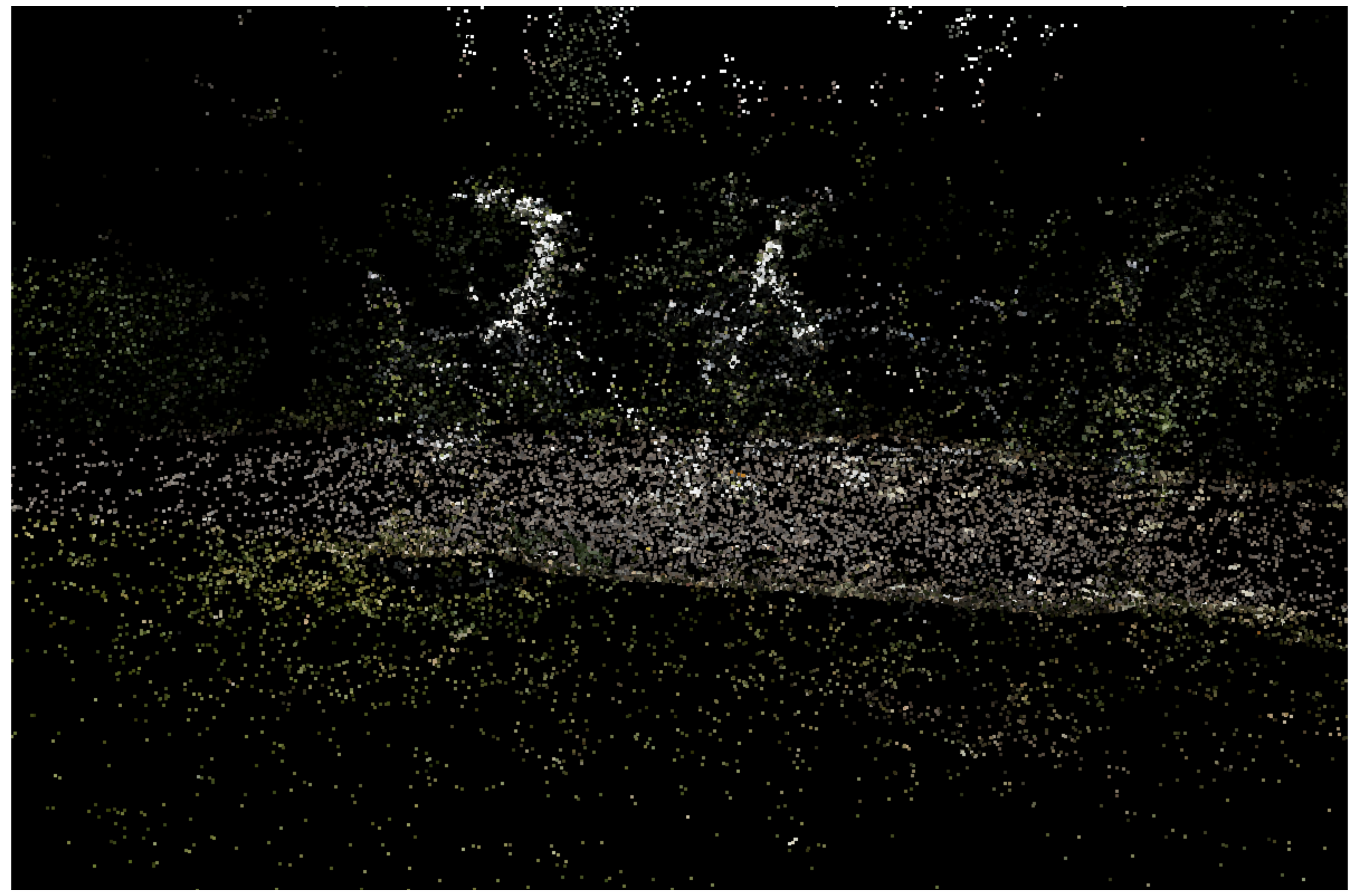


Rasterization

Memory footprint of 3DGS

High resolution 3D scene
(1600 x 1063)

3D Gaussian
representation



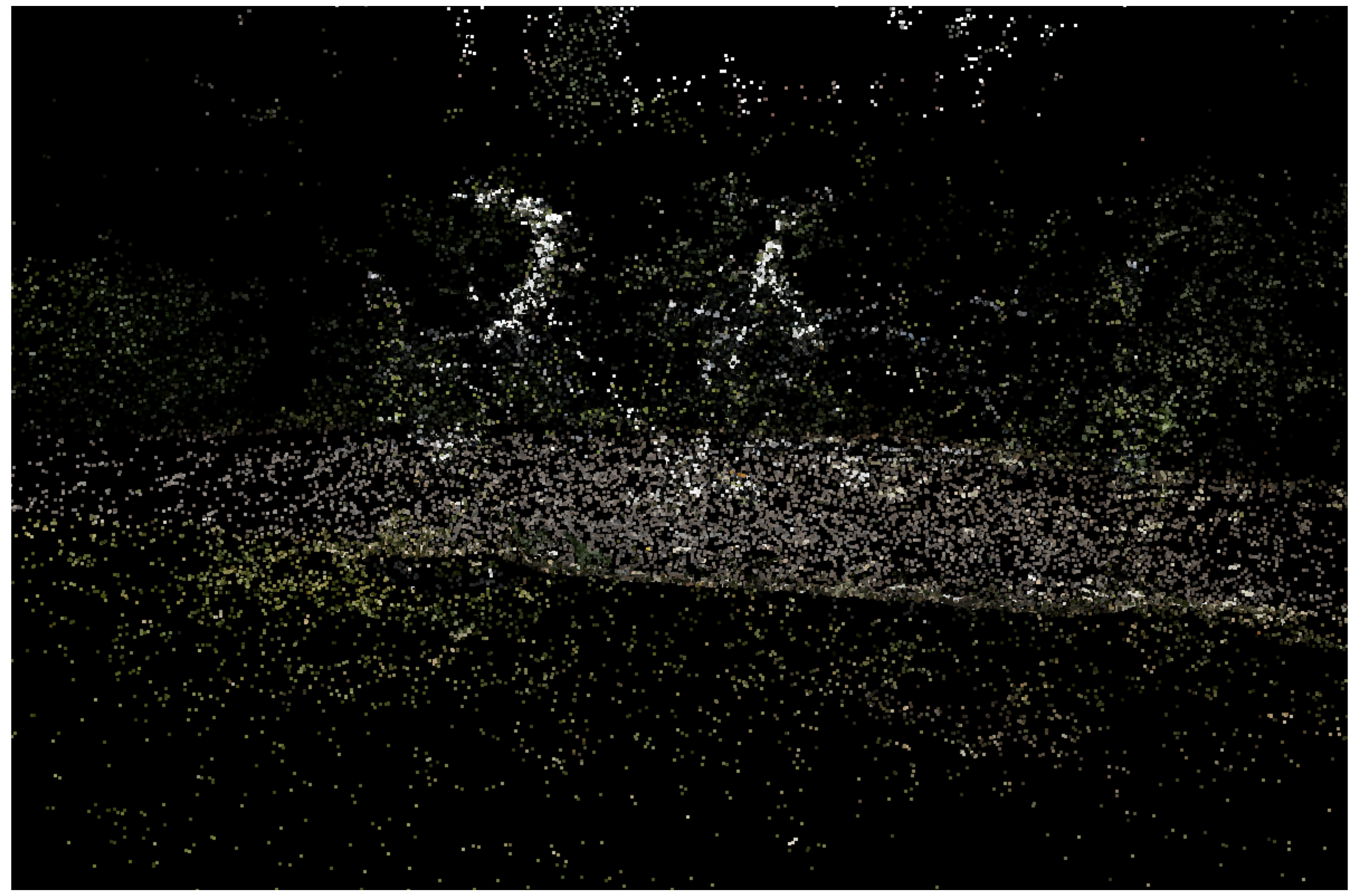
5 million Gaussians

Memory footprint of 3DGS

High resolution 3D scene
(1600 x 1063)

3D Gaussian

representation



5 million Gaussians

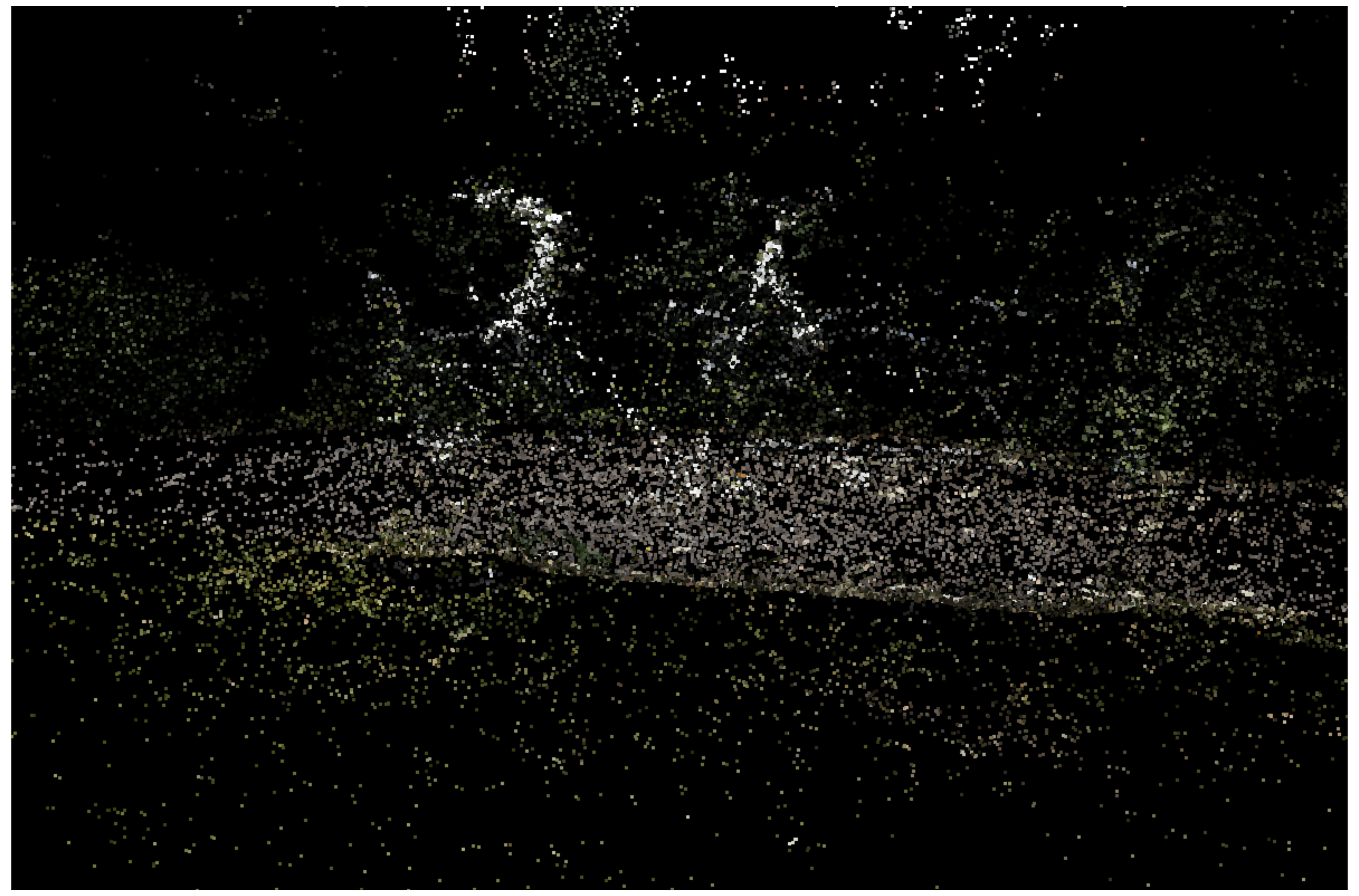
Per Gaussian memory: 0.25 KB

Total storage memory: 1.25 GB

Memory footprint of 3DGS

High resolution 3D scene
(1600 x 1063)

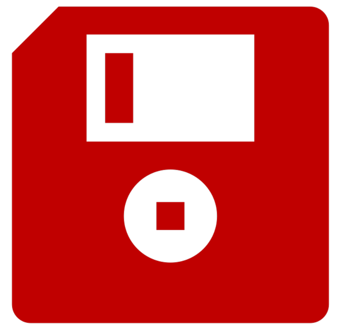
3D Gaussian representation



5 million Gaussians

Per Gaussian memory: 0.25 KB

Total storage memory: 1.25 GB



High per Gaussian memory cost

Large number of Gaussians

Improving Efficiency Metrics

Improving Efficiency Metrics

- Memory efficiency
  - Training and inference runtime memory (GPU RAM)
  - Post-training storage memory

Improving Efficiency Metrics

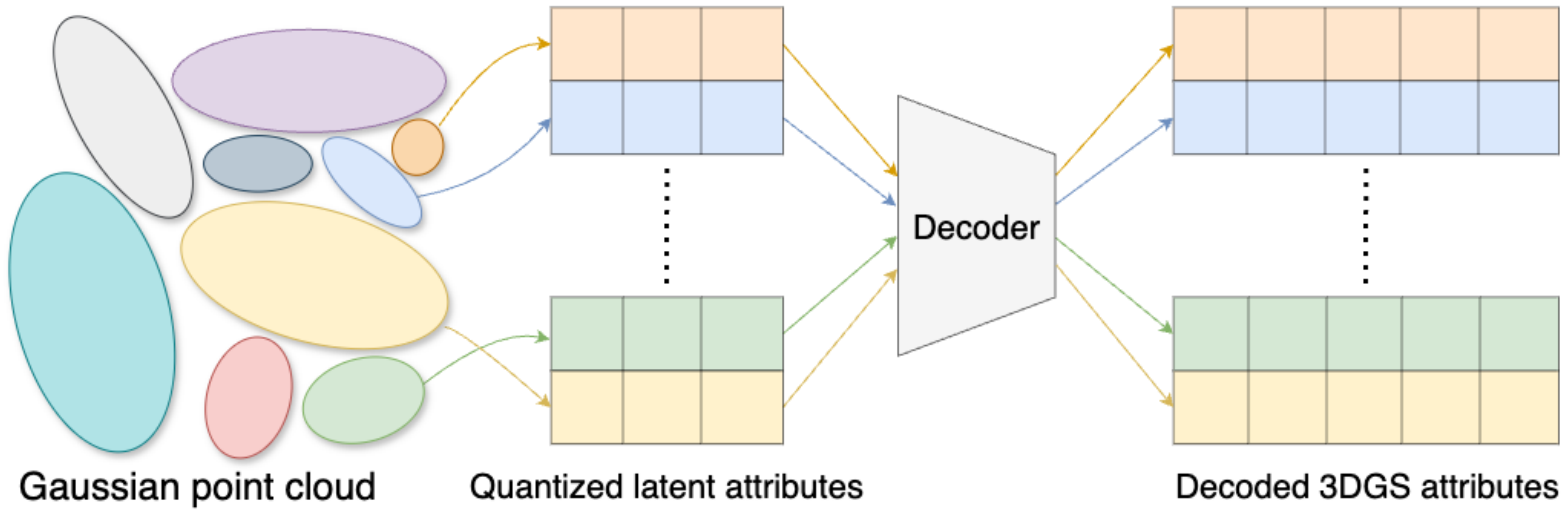
- Memory efficiency
  - Training and inference runtime memory (GPU RAM)
  - Post-training storage memory
- Time efficiency
  - Training time
  - Rendering speed

Improving Efficiency Metrics

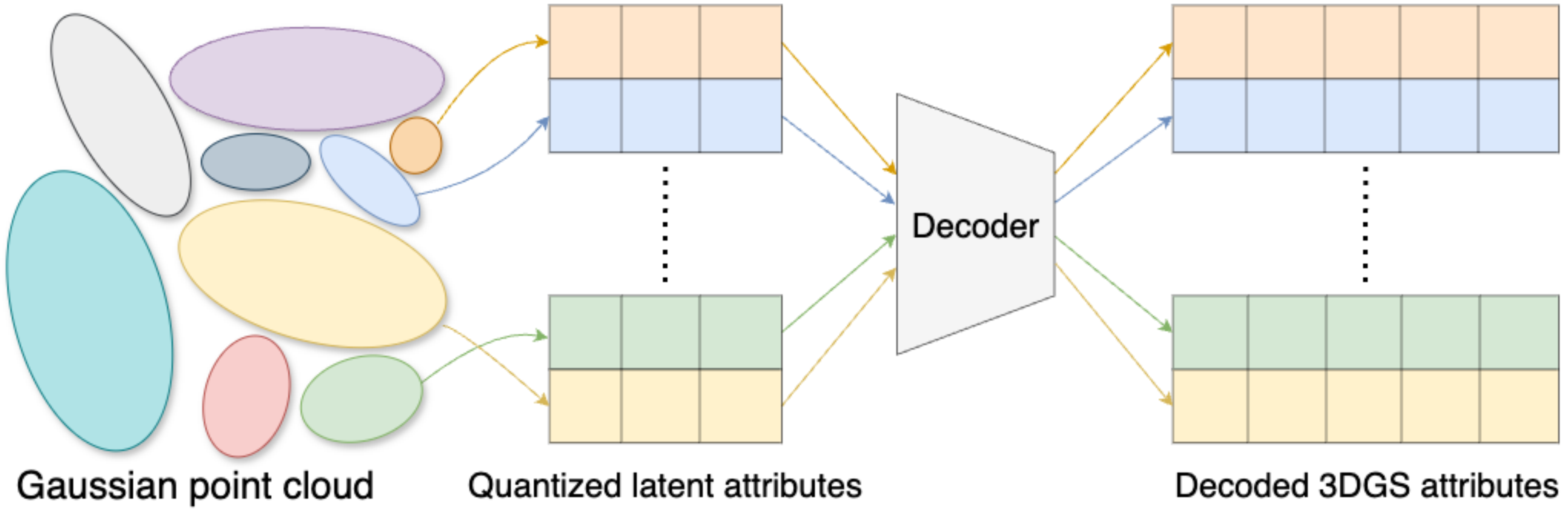
- Memory efficiency
    - Training and inference runtime memory (GPU RAM)
    - Post-training storage memory
- Time efficiency
    - Training time
    - Rendering speed

While maintaining reconstruction quality!

EAGLES: Efficient 3D Gaussian splatting



EAGLES: Efficient 3D Gaussian splatting



1. Quantize attributes for reducing memory storage costs of each Gaussian point.

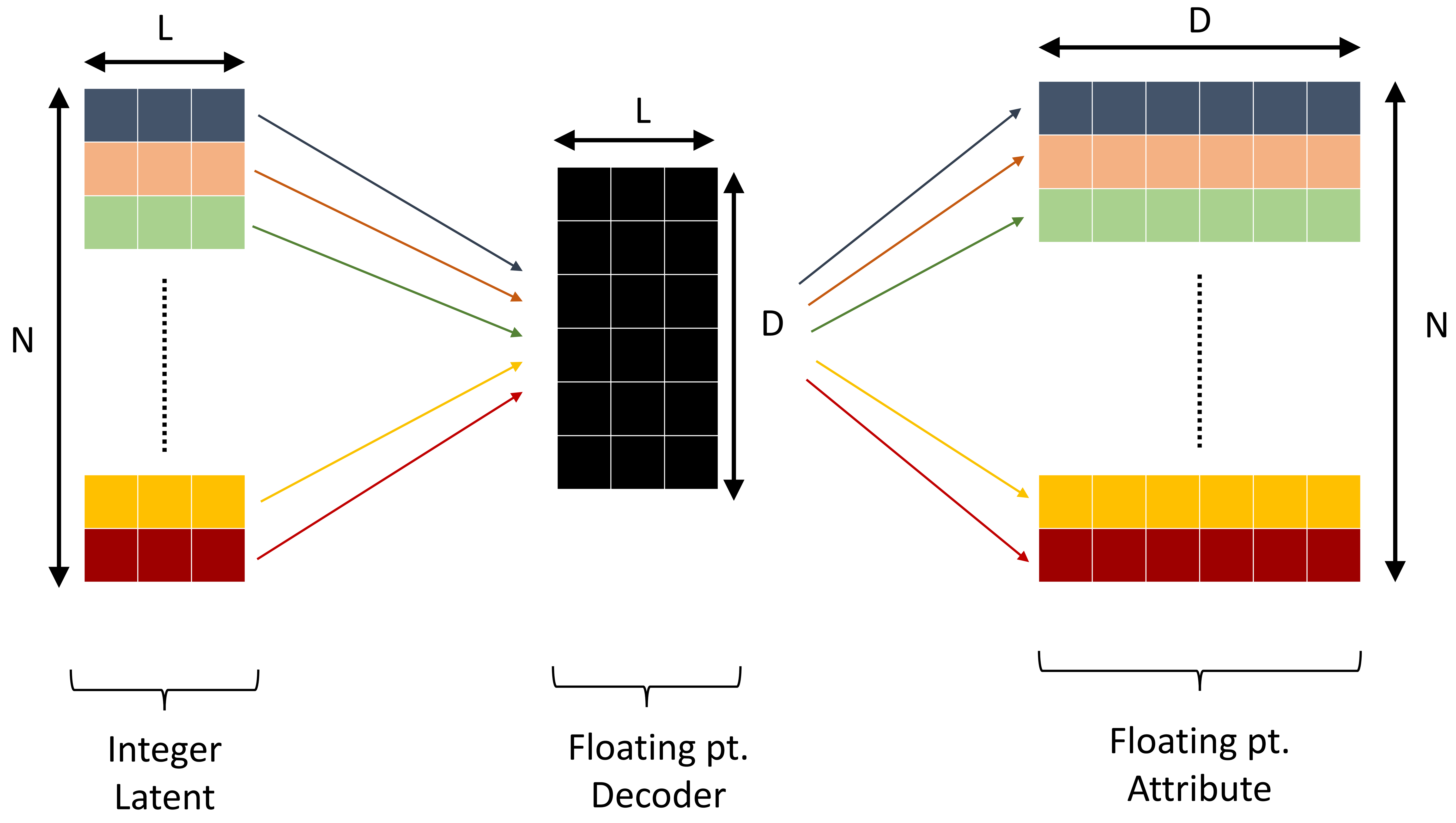
EAGLES: Efficient 3D Gaussian splatting

1. Quantize attributes for reducing memory storage costs of each Gaussian point.
2. Utilize a coarse-to-fine training strategy for faster training and stable optimization of the Gaussians.

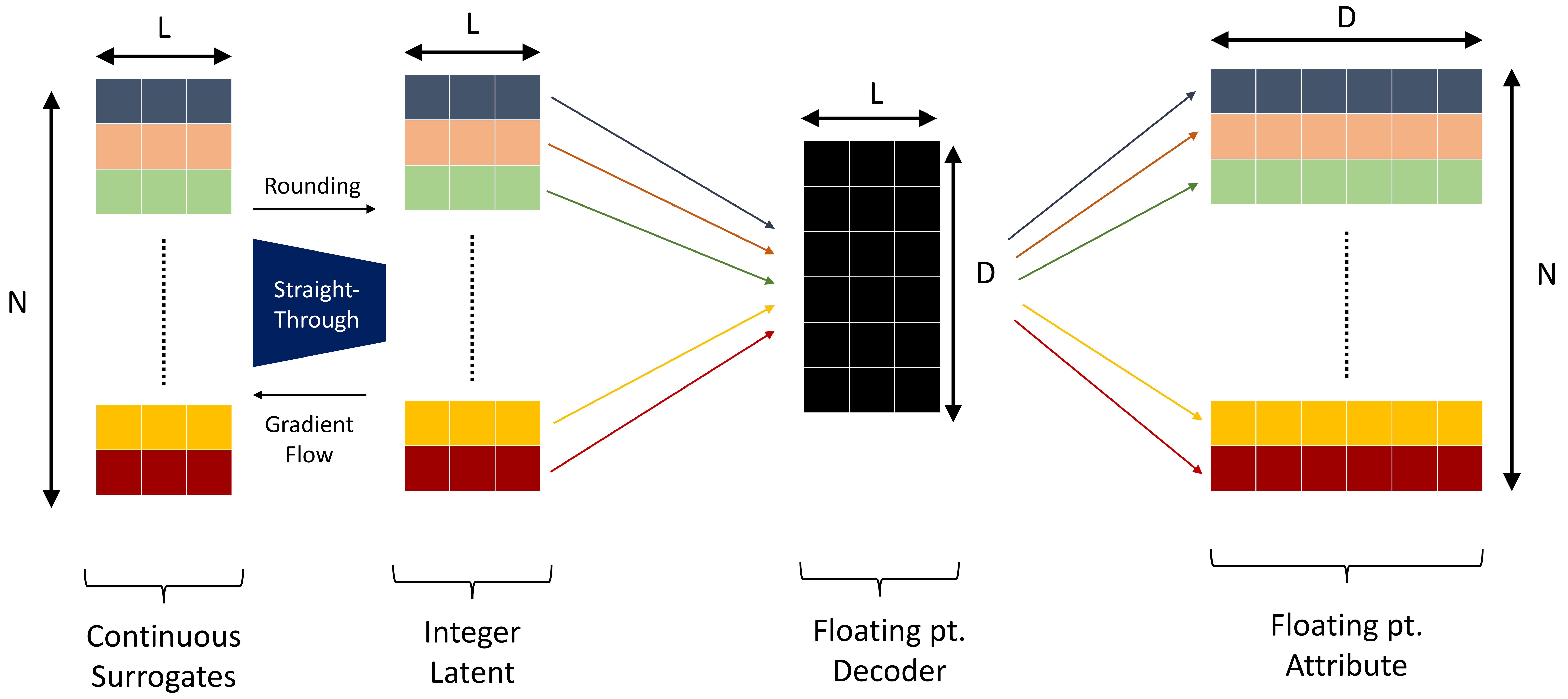
EAGLES: Efficient 3D Gaussian splatting

1. Quantize attributes for reducing memory storage costs of each Gaussian point.
2. Utilize a coarse-to-fine training strategy for faster training and stable optimization of the Gaussians.
3. Perform pruning of redundant Gaussians leading to fewer points for storage and rendering.

Attribute quantization



Attribute quantization

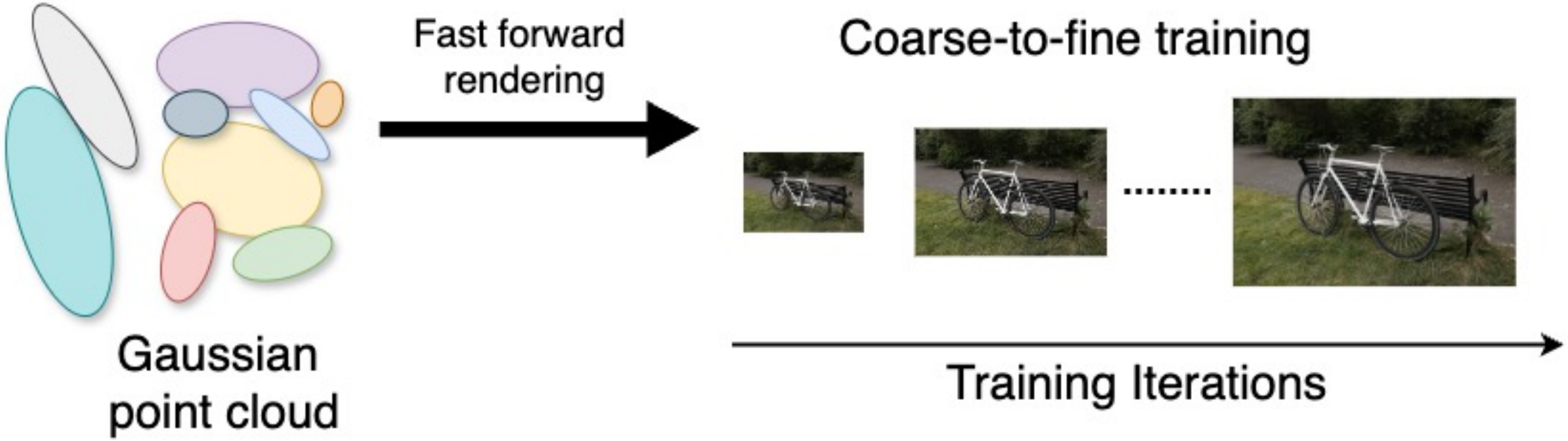


End-to-end differentiable with decoder and straight-through estimator

Qauantization aware training instead of post-training quantization recovers performance losses.

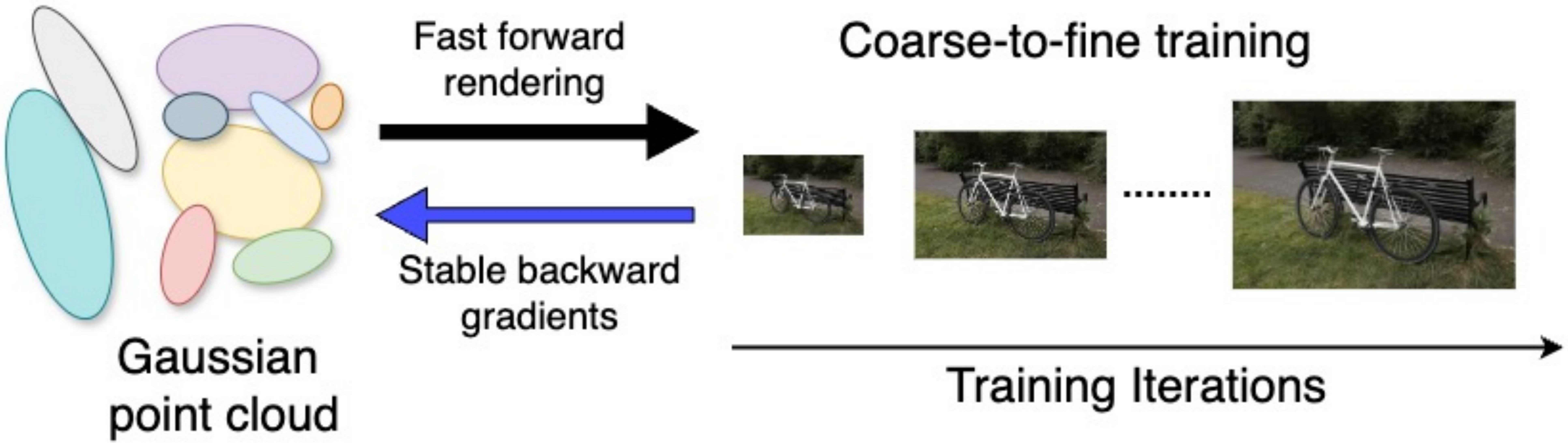
Coarse-to-fine training

Render images at progressively increasing resolution during training



Coarse-to-fine training

Stable optimization along with faster convergence



Coarse-to-fine training

Gaussian pruning

Large number of redundant Gaussians

↓

Define importance/saliency score for Gaussians on scene rendering

↓

Remove least influential/important Gaussians

Influence score

$$C = \sum_{i \in \mathcal{N}} \boldsymbol{c}_i \alpha_i \underbrace{\prod_{j=1}^{i-1} (1 - \alpha_j)}$$

Rasterization equation

$$W_{\mathrm{i,p}} = \alpha_i T_i = \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad W_{\mathrm{i}} = \sum_p W_{\mathrm{i,p}}$$

Influence of Gaussian i at pixel p

No computation overhead

Net influence of Gaussian i

Prune lowest s% Gaussians

Influence score

$$W_{\mathrm{i,p}} = \alpha_i T_i = \alpha_i \prod_{j=1}^{i-1}(1 - \alpha_j), \quad W_{\mathrm{i}} = \sum_p W_{\mathrm{i,p}}$$

Low influence for transparent Gaussians (low opacity)

Influence score

$$W_{\mathrm{i,p}} = \alpha_i T_i = \alpha_i \underbrace{\prod_{j=1}^{i-1}(1-\alpha_j),}_{\text{Low influence for occluded Gaussians (low transmittance)}} \quad W_{\mathrm{i}} = \sum_p W_{\mathrm{i,p}}$$

Influence score

$$W_{\mathrm{i,p}} = \alpha_i T_i = \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \quad W_{\mathrm{i}} = \sum_p W_{\mathrm{i,p}}$$

Low influence for small Gaussians affecting fewer pixels (low scaling attribute)

Influence pruning

Quantitative results

Dataset (→) Method	Mip-NeRF360 PSNR ↑	SSIM ↑	LPIPS ↓	Storage Mem ↓	FPS ↑	Train Time ↓
Plenoxels	23.08	0.63	0.46	2.1GB	7	25m49s
INGP	25.59	0.70	0.33	48MB	9	7m30s
M-NeRF360	27.69	0.79	0.24	9MB	0.06	48h
3D-GS	27.21	0.82	0.21	734MB	134	41m33s
3D-GS*	27.45	0.81	0.22	745MB	110	23m20s
EAGLES (Ours)	27.23	0.81	0.24	54MB	131	21m34s
EAGLES-Small	26.94	0.80	0.25	47MB	166	17m3s
EAGLES-Fast	26.99	0.81	0.23	71MB	111	16m24s

Quantitative results

Dataset (→)	Mip-NeRF360					
Method	PSNR ↑	SSIM ↑	LPIPS ↓	Storage Mem ↓	FPS ↑	Train Time ↓
Plenoxels	23.08	0.63	0.46	2.1GB	7	25m49s
INGP	25.59	0.70	0.33	48MB	9	7m30s
M-NeRF360	27.69	0.79	0.24	9MB	0.06	48h
3D-GS	27.21	0.82	0.21	734MB	134	41m33s
3D-GS*	27.45	0.81	0.22	745MB	110	23m20s
EAGLES (Ours)	27.23	0.81	0.24	54MB	131	21m34s
EAGLES-Small	26.94	0.80	0.25	47MB	166	17m3s
EAGLES-Fast	26.99	0.81	0.23	71MB	111	16m24s

Similar reconstruction quality as 3D-GS

Quantitative results

Dataset (→)	Mip-NeRF360					
Method	PSNR ↑	SSIM ↑	LPIPS ↓	Storage Mem ↓	FPS ↑	Train Time ↓
Plenoxels	23.08	0.63	0.46	2.1GB	7	25m49s
INGP	25.59	0.70	0.33	48MB	9	7m30s
M-NeRF360	27.69	0.79	0.24	9MB	0.06	48h
3D-GS	27.21	0.82	0.21	734MB	134	41m33s
3D-GS*	27.45	0.81	0.22	745MB	110	23m20s
EAGLES (Ours)	27.23	0.81	0.24	54MB	131	21m34s
EAGLES-Small	26.94	0.80	0.25	47MB	166	17m3s
EAGLES-Fast	26.99	0.81	0.23	71MB	111	16m24s

14x reduction in storage memory

Quantitative results

Dataset (→)	Mip-NeRF360					
Method	PSNR ↑	SSIM ↑	LPIPS ↓	Storage Mem ↓	FPS ↑	Train Time ↓
Plenoxels	23.08	0.63	0.46	2.1GB	7	25m49s
INGP	25.59	0.70	0.33	48MB	9	7m30s
M-NeRF360	27.69	0.79	0.24	9MB	0.06	48h
3D-GS	27.21	0.82	0.21	734MB	134	41m33s
3D-GS*	27.45	0.81	0.22	745MB	110	23m20s
EAGLES (Ours)	27.23	0.81	0.24	54MB	131	21m34s
EAGLES-Small	26.94	0.80	0.25	47MB	166	17m3s
EAGLES-Fast	26.99	0.81	0.23	71MB	111	16m24s

Lower training time and higher rendering speed

Quantitative results

Dataset (→)	Mip-NeRF360						Tanks&Temples						Deep Blending					
Method	PSNR ↑	SSIM ↑	LPIPS ↓	Storage Mem ↓	FPS ↑	Train Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Storage Mem ↓	FPS ↑	Train Time ↓	PSNR ↑	SSIM ↑	LPIPS ↓	Storage Mem ↓	FPS ↑	Train Time ↓
Plenoxels	23.08	0.63	0.46	2.1GB	7	25m49s	21.08	0.72	0.38	2.3GB	13	25m5s	23.06	0.80	0.51	2.7GB	11	27m49s
INGP	25.59	0.70	0.33	48MB	9	7m30s	21.92	0.75	0.31	48MB	14	6m59s	24.96	0.82	0.39	48MB	3	8m
M-NeRF360	27.69	0.79	0.24	9MB	0.06	48h	22.22	0.76	0.26	9MB	0.14	48h	29.40	0.90	0.25	8.6MB	0.09	48h
3D-GS	27.21	0.82	0.21	734MB	134	41m33s	23.61	0.84	0.18	411MB	154	26m54s	29.41	0.90	0.24	676MB	137	36m2s
3D-GS*	27.45	0.81	0.22	745MB	110	23m20s	23.63	0.85	0.18	430MB	157	12m5s	29.55	0.90	0.25	656MB	123	23m5s
EAGLES (Ours)	27.23	0.81	0.24	54MB	131	21m34s	23.37	0.84	0.20	29MB	227	11m39s	29.86	0.91	0.25	52MB	130	21m50s
EAGLES-Small	26.94	0.80	0.25	47MB	166	17m3s	23.10	0.82	0.22	19MB	272	10m7s	29.92	0.90	0.25	33MB	160	17m40s
EAGLES-Fast	26.99	0.81	0.23	71MB	111	16m24s	23.02	0.83	0.20	38MB	190	8m43s	29.85	0.91	0.25	63MB	108	16m30s

Similar results for wide variety of datasets

Marginally outperforms 3D-GS in reconstruction quality

Reduced GPU runtime memory

Method	Bicycle		Truck		Playroom	
	Train	Render	Train	Render	Train	Render
3D-GS	17.4G	9.5G	8.5G	4.8G	9.6G	6.0G
EAGLES	10G	7.4G	5.3G	3.6G	7.1G	5.3G

Training and inference runtime memory is lower due to fewer number of Gaussians.

Qualitative results

Conclusion

- We compress per-point Gaussian attributes via an end-to-end learnable latent quantization framework.
- We introduce coarse-to-fine training to improve optimization stability of 3DGS while speeding up training.
- We develop a pruning stage to reduce redundant/insignificant Gaussians for lower memory and higher rendering speeds.

Visit us at poster session 6 on Thursday evening (16:30 – 18:30) at ECCV 2024!

Project page with code:

Thank you