Qualcomm

ICLR
International Conference On Learning Representations

Sort-free Gaussian Splatting via Weighted Sum Rendering

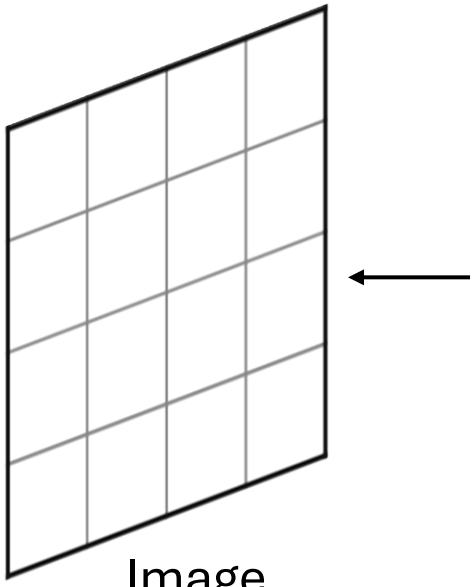
Qiqi Hou [1], Randall Rauwendaal [2], Zifeng Li [2], Hoang Le [1], Farzad Farhadzadeh [1], Fatih Porikli [1], Alexei Bourd [2], Amir Said[*1]

[1] Qualcomm AI Research[†] [2] Graphic Research Team

* Corresponding author
† Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc

Motivation

Qualcomm AI Research



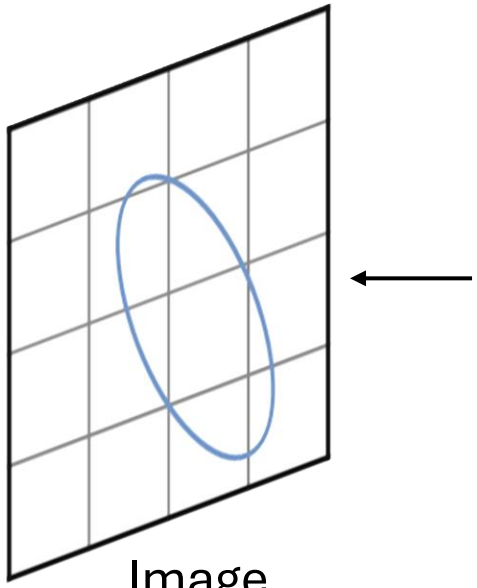
Image

Gaussians

Gaussian Splatting

Motivation

Qualcomm AI Research



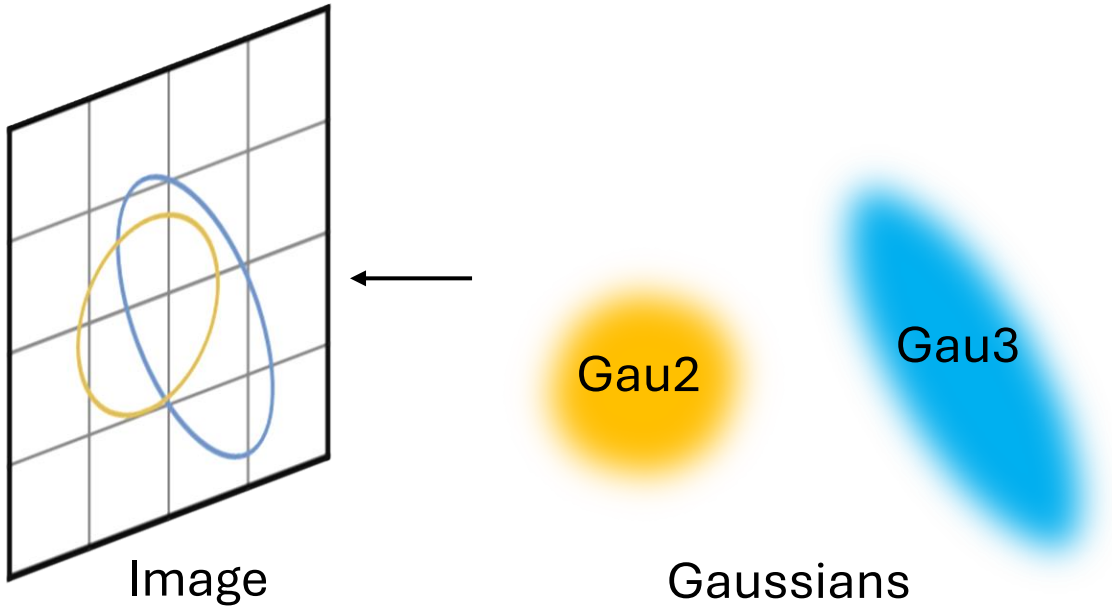
Image

Gaussians

Gaussian Splatting

Motivation

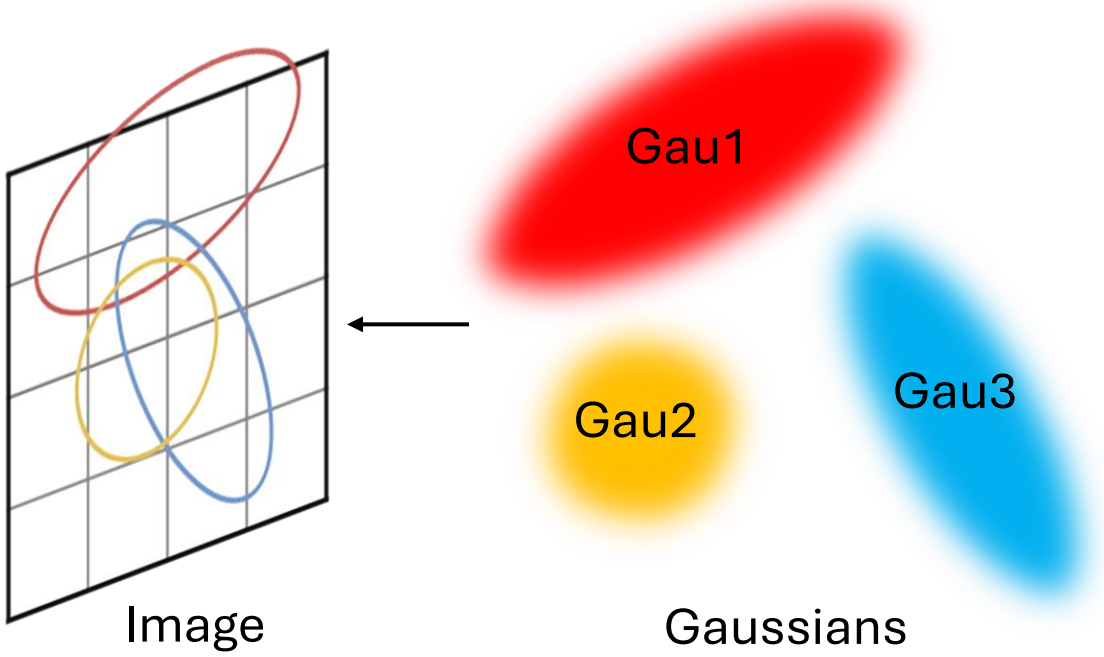
Qualcomm AI Research



Gaussian Splatting

Motivation

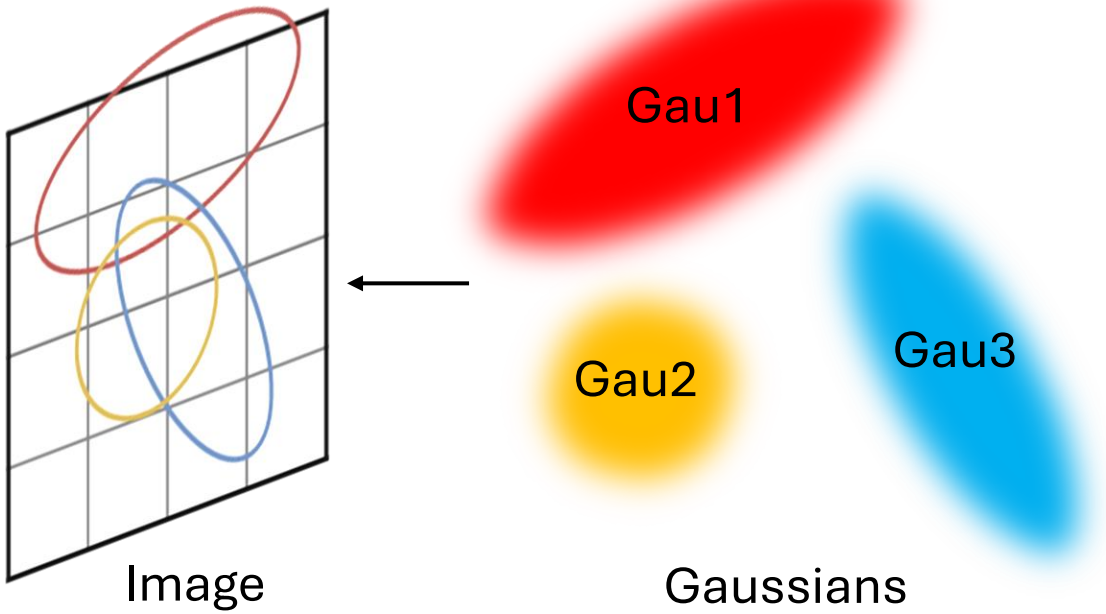
Qualcomm AI Research

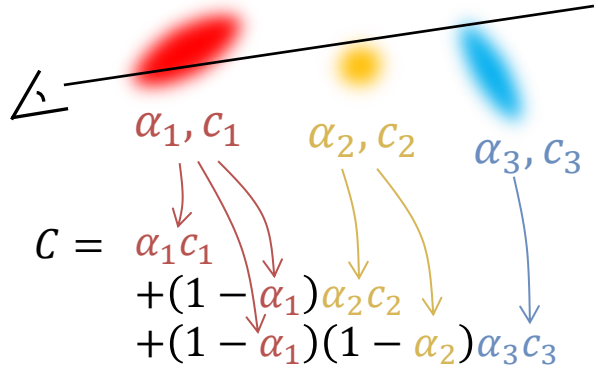


Gaussian Splatting

Motivation

Qualcomm AI Research

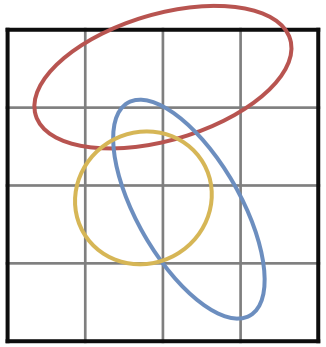




Gaussian Splatting

Motivation

Qualcomm AI Research

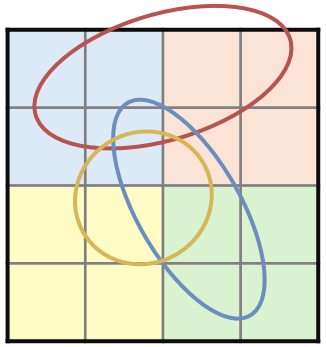


Tiling

Gaussian Splatting Rendering

3DGS splits the rendered image into multiple non-overlapping tiles.

Motivation



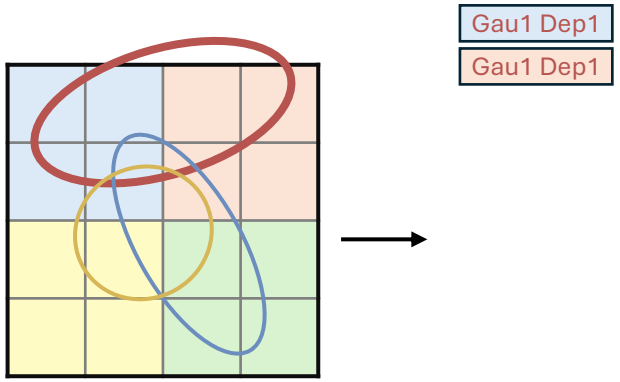
Tiling

e.g. 16 x 16 pixels

Tiling

3DGS splits the rendered image into multiple non-overlapping tiles.

Motivation

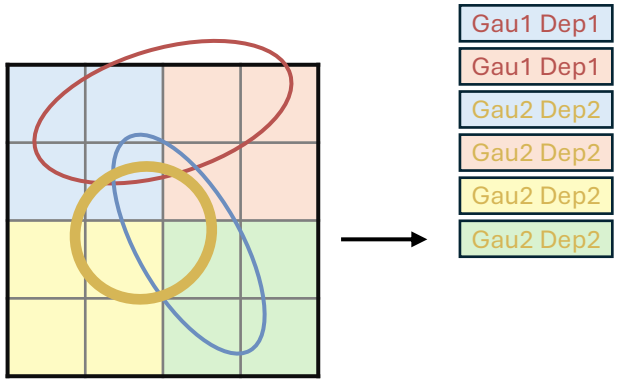


Tiling Replication

Replication

Gaussians that span across multiple tiles being duplicated accordingly

Motivation



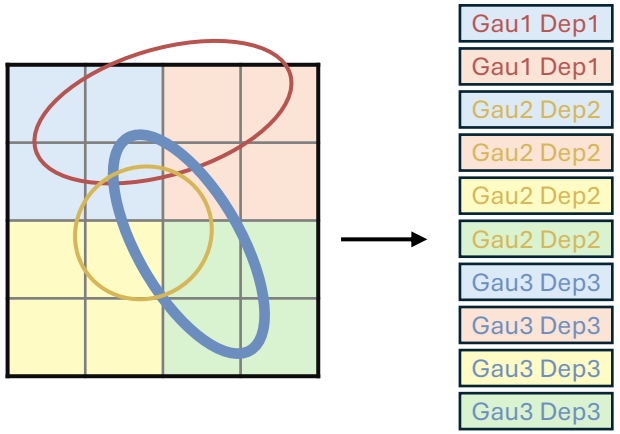
Tiling Replication

Replication

Gaussians that span across multiple tiles being duplicated accordingly

Motivation

Qualcomm AI Research

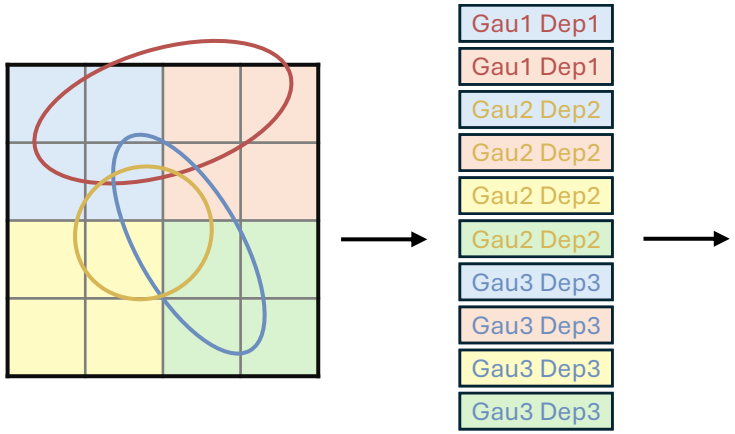


Tiling Replication

Replication

Gaussians that span across multiple tiles being duplicated accordingly

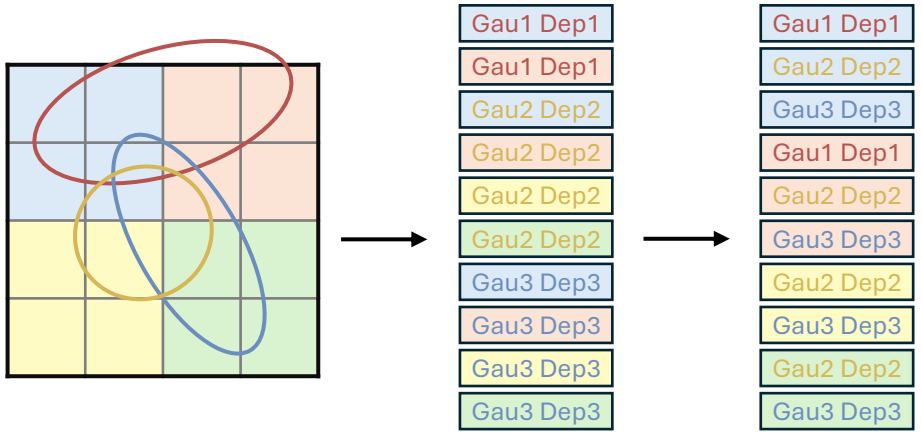
Motivation



Sorting

3DGS sort the Gaussians according to the depth.

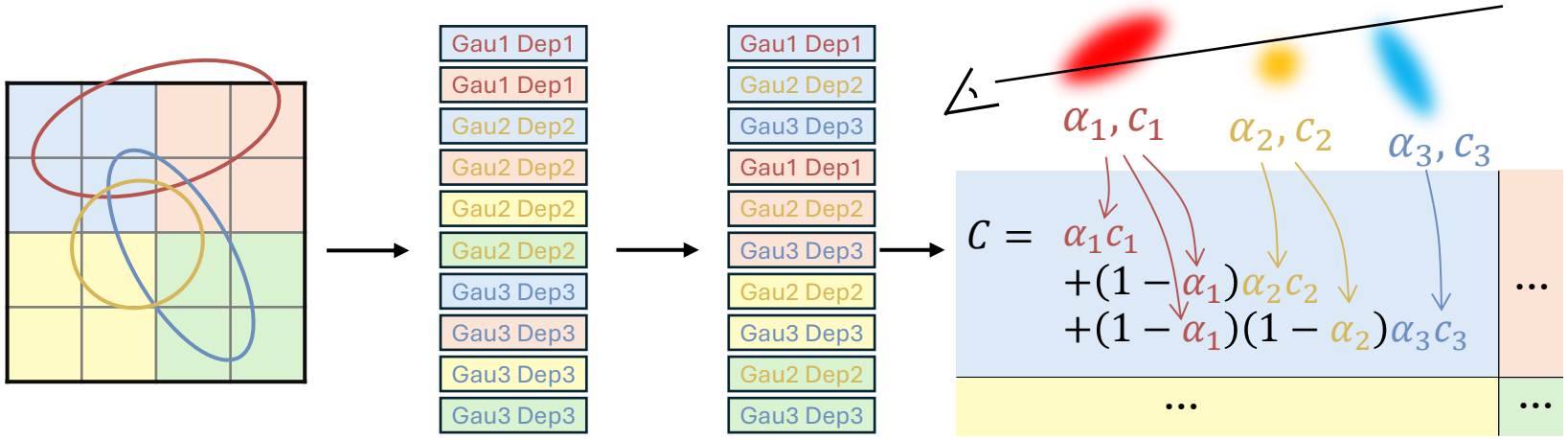
Motivation



Sorting

3DGS sort the Gaussians according to the depth.

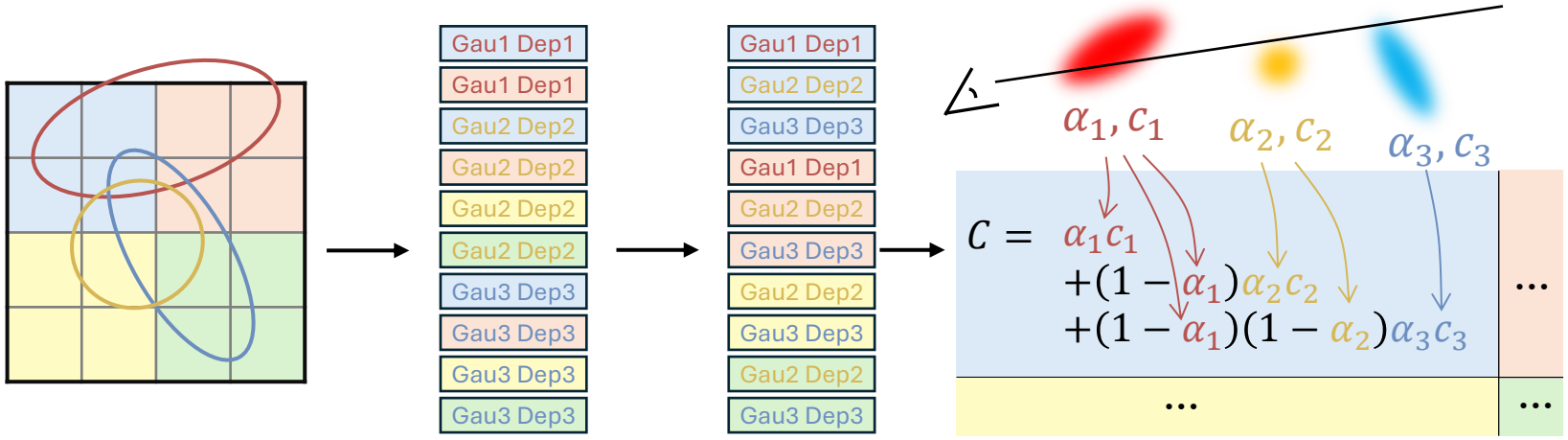
Motivation



Alpha blending

3DGS relies on non-commutative alpha blending.

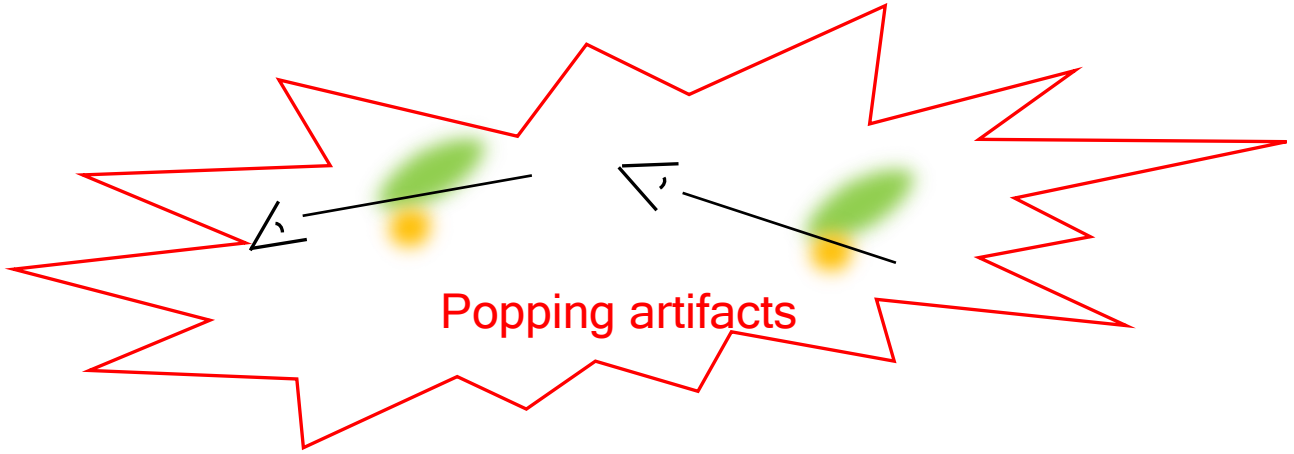
Motivation

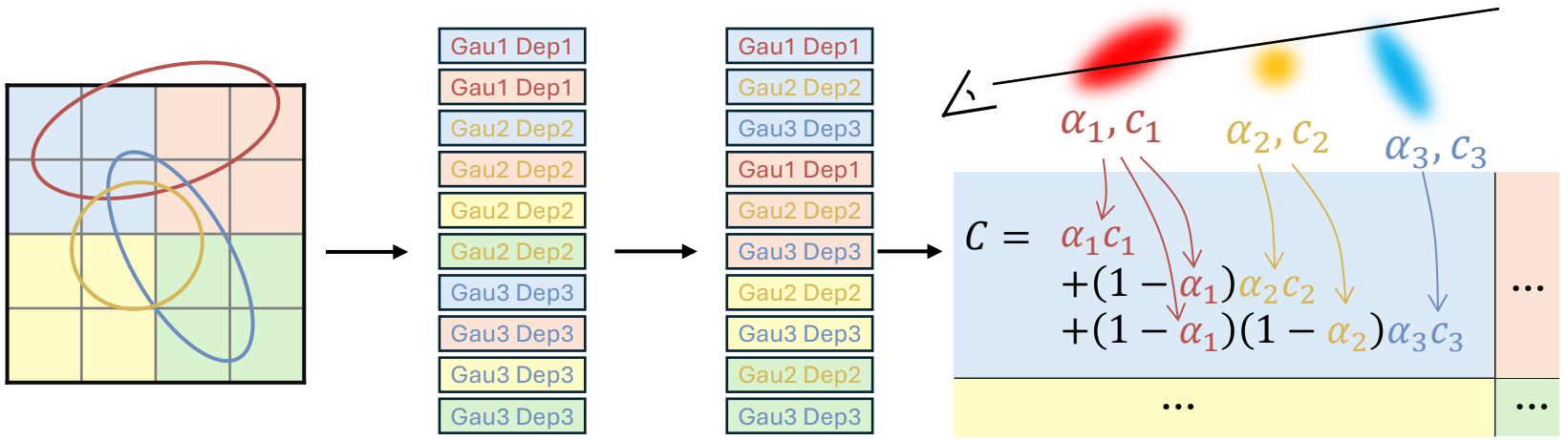


Sorting is not friendly for GPU Hardware Rasterization

It introduces computational overhead and memory bottlenecks.

Motivation



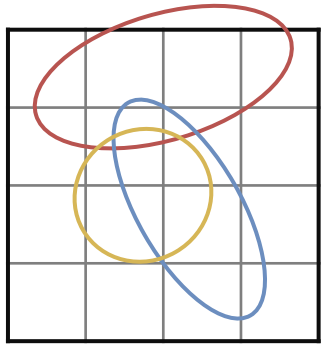


Tiling Replication **Sorting** Rendering

Sorting introduces popping artifacts

Our method

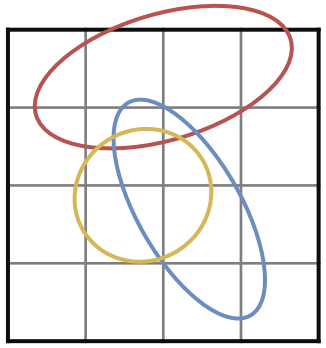
Qualcomm AI Research



Can we remove sorting in 3DGS?

Our method

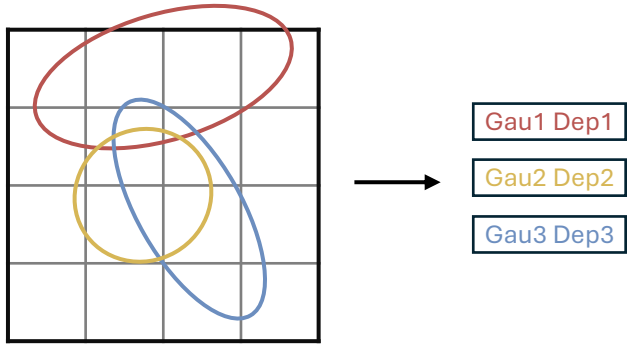
Qualcomm AI Research



Sort-free Gaussian Splatting

Our method

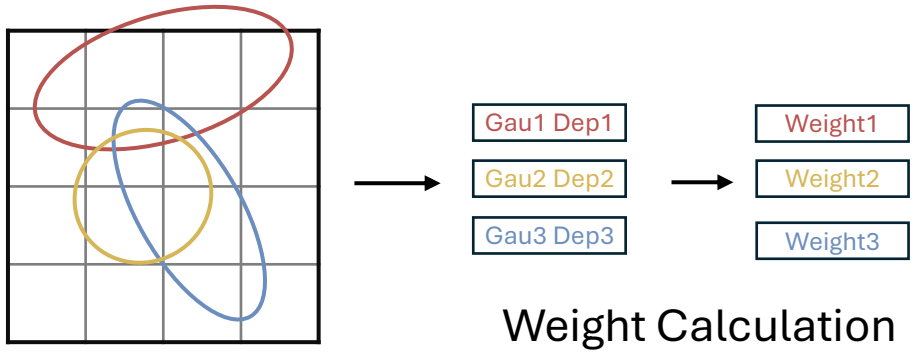
Qualcomm AI Research



Sort-free Gaussian Splatting

Our method

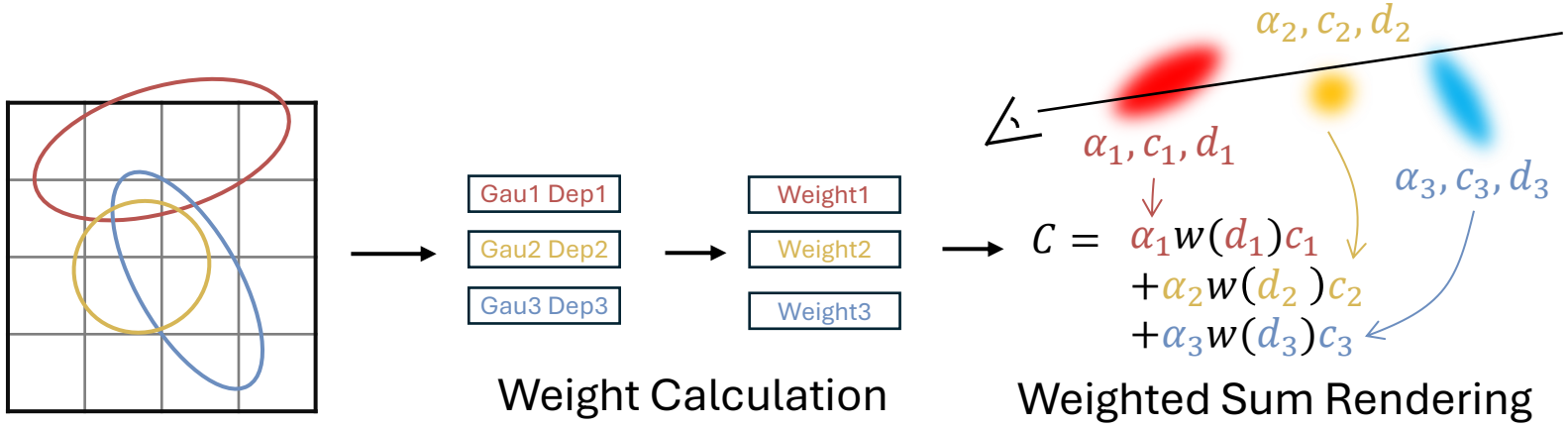
Qualcomm AI Research



Sort-free Gaussian Splatting

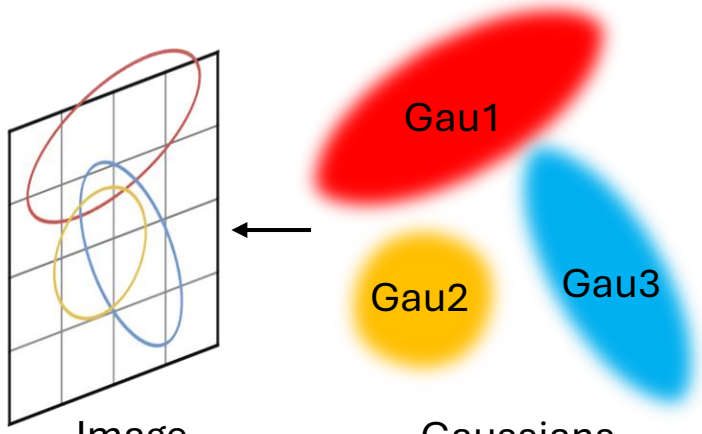
Our method

Qualcomm AI Research

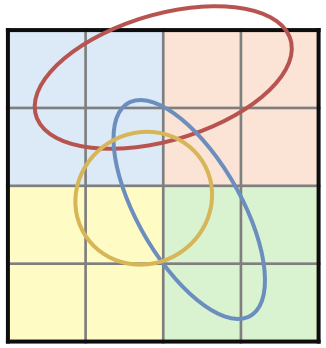


Sort-free Gaussian Splatting

Our method

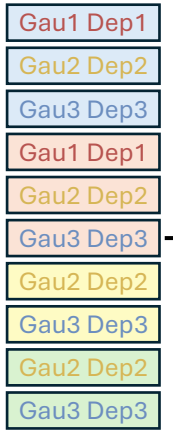


Splatting

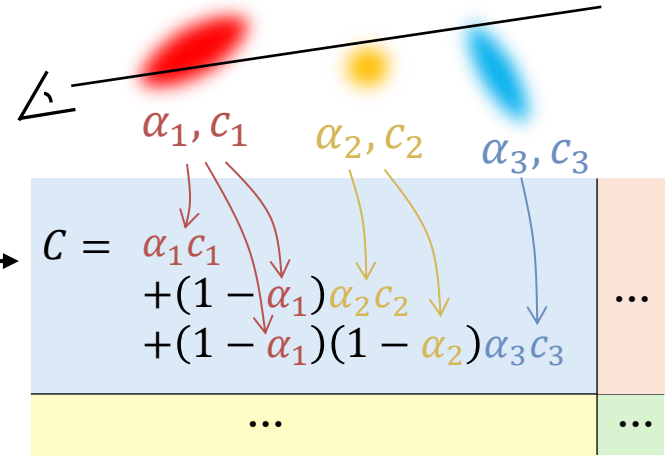


Tiling

Replication



Sorting



Rendering

3DGS with sort

Sort-free Gaussian Splatting via Weighted Sum Rendering

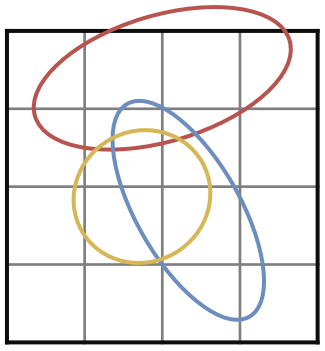
Gaussians

Pixel

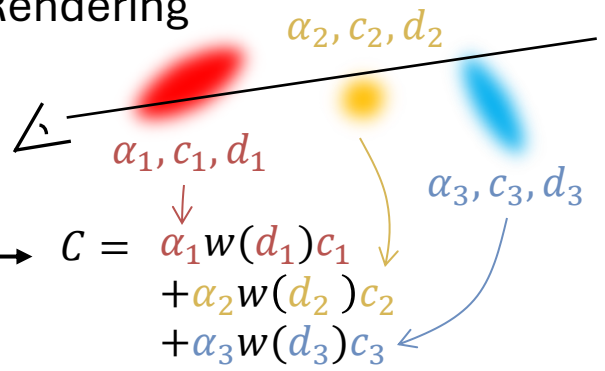
Tiles

Gaussian 1 for the blue tile with depth 1

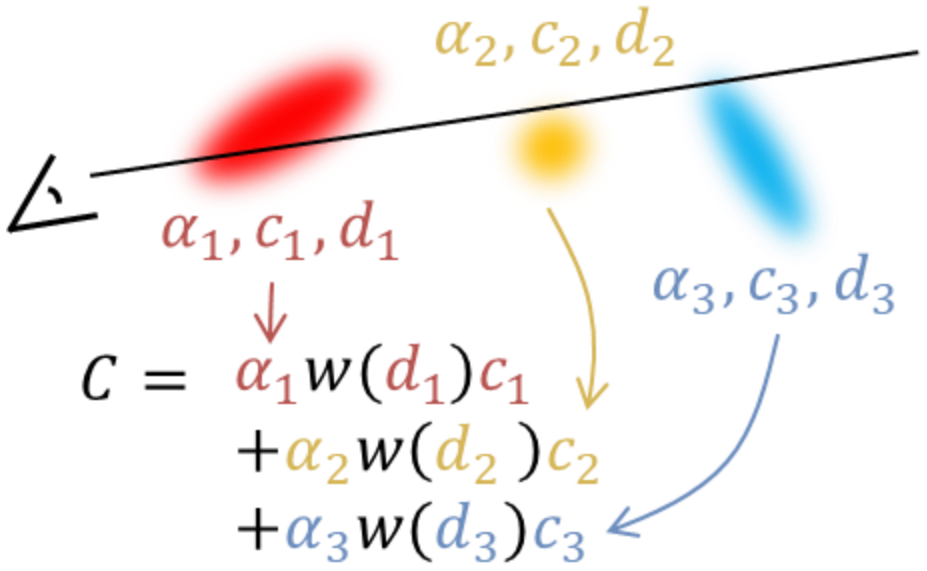
α Alpha c Color d Depth



Weight Calculation



Weighted Sum Rendering



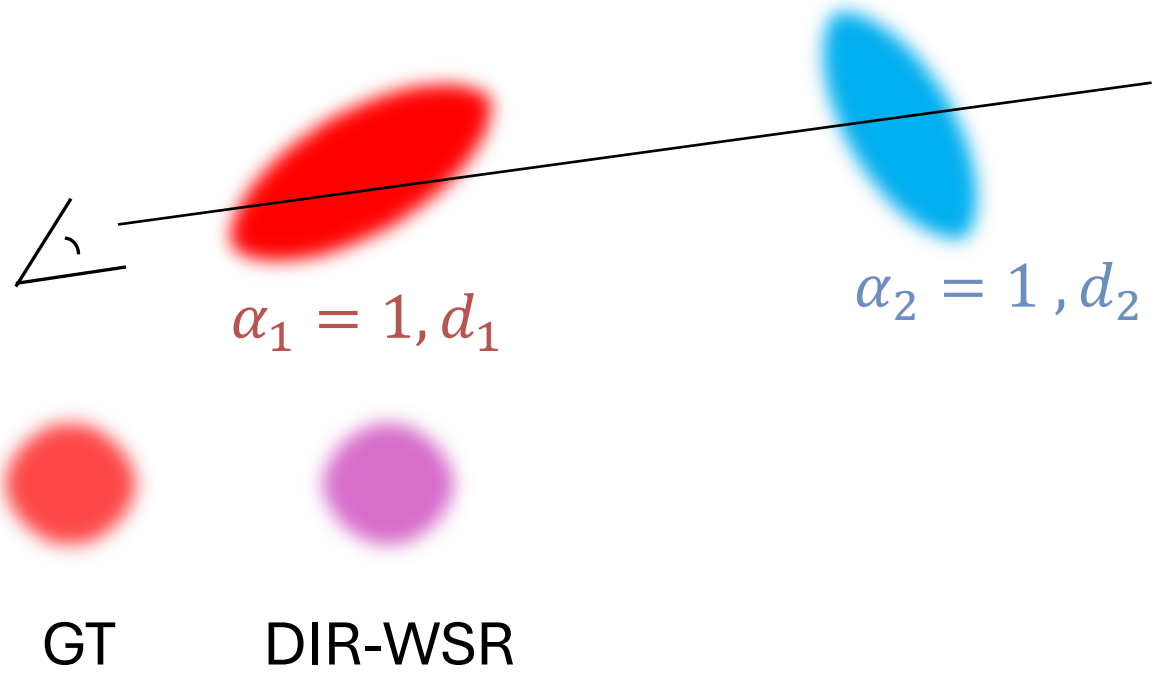
$$\mathbf{C} = \frac{\mathbf{c}_B w_B + \sum_{i=1}^{N} \mathbf{c}_i \alpha_i w(d_i)}{w_B + \sum_{i=1}^{N} \alpha_i w(d_i)}$$

Weighted Sum Rendering

How to define the weight function w(d), and alpha?

WSR

Qualcomm AI Research



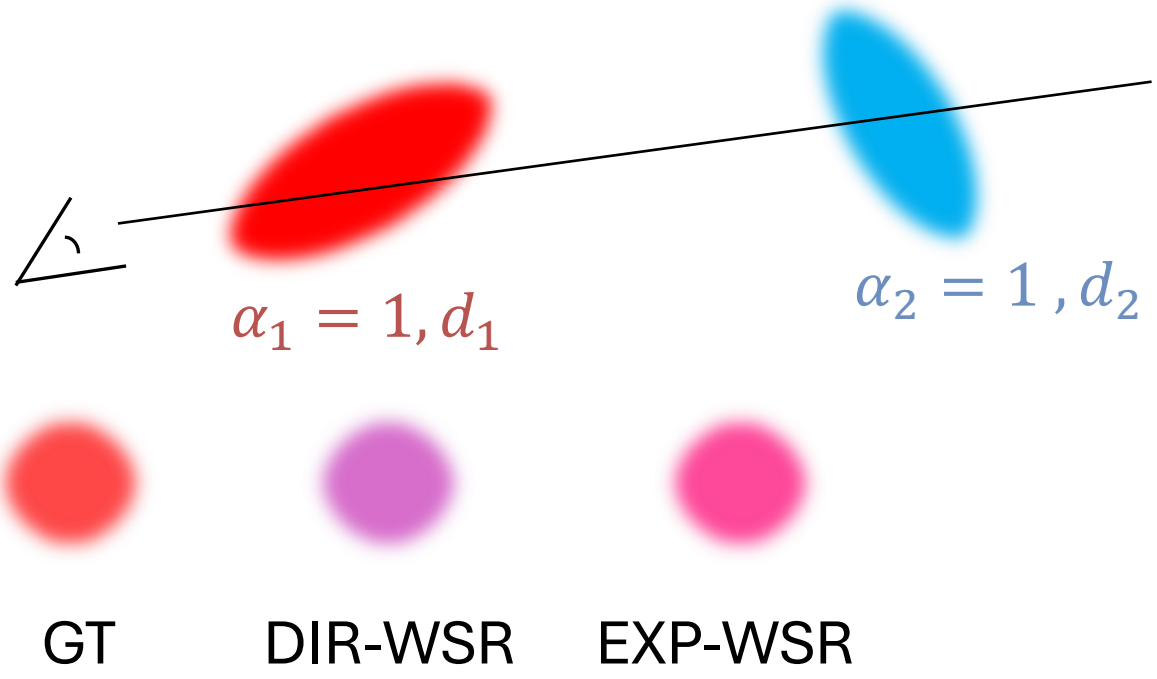
$$w(d_i) = 1$$

Direct Weighted Sum Rendering

No learnable parameter.

WSR

Qualcomm AI Research



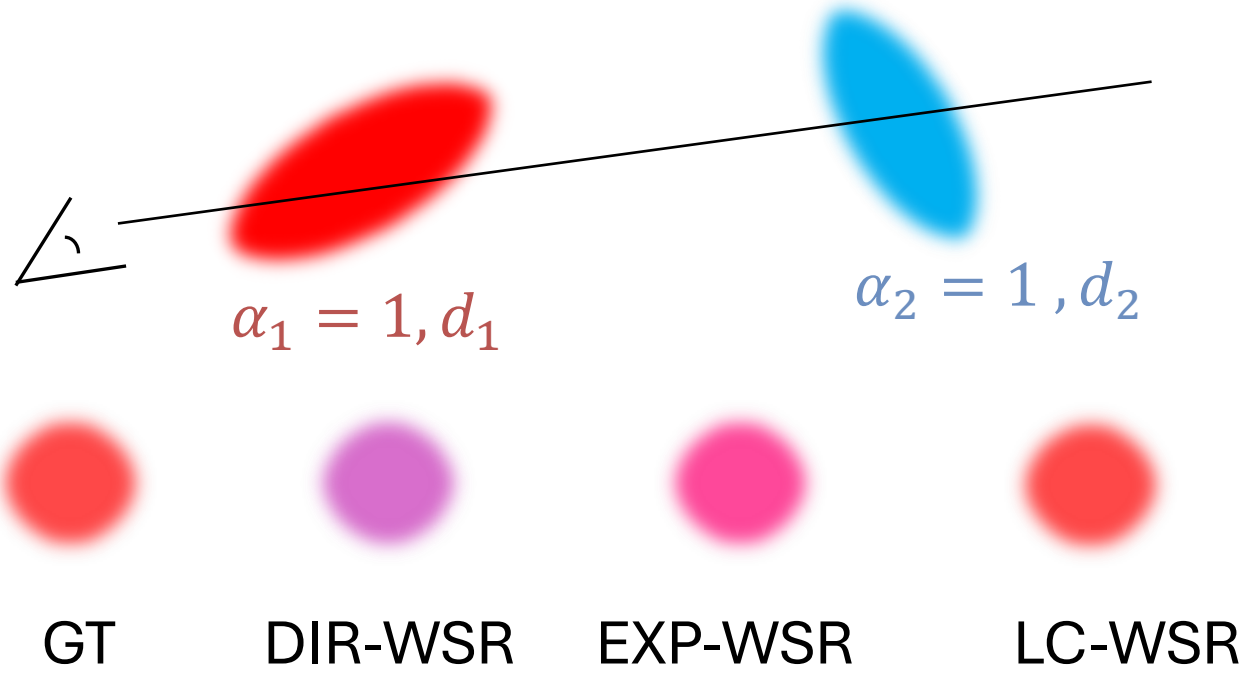
$$w(d_i) = \exp(-\sigma d_i^{\beta})$$

Exponential Weighted Sum Rendering

Learnable parameters σ and β enable to produce more accurate results.

WSR

Qualcomm AI Research

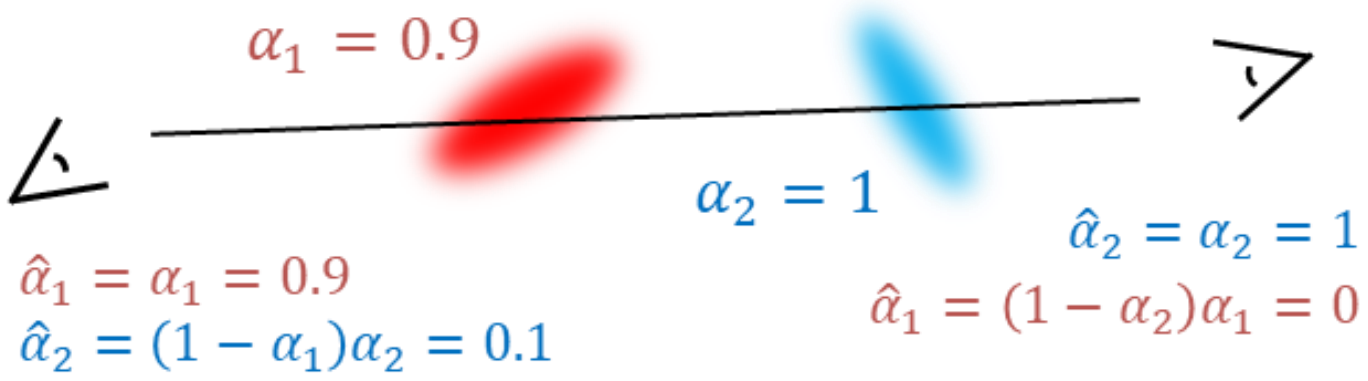


$$w(d_i) = \max\left(0, 1 - \frac{d_i}{\sigma}\right) v_i$$

Linear Correction Weighted Sum Rendering

ReLU like function. Never used in OIT. We are the first to use it in rendering.

View dependent opacity



$$u_i = Y(\|\mathbf{f} - \mathbf{p}_i\|, \mathbf{H}_i)$$

View dependent opacity

Y (·) indicates the spherical harmonic function and $\mathbf{H}_i$ indicates the learnable spherical harmonics coefficients for opacity.

Experiments

Qualcomm AI Research

Method	Mip-NeRF 360			Tanks & Temples			Deep Blending		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Plenoxels	23.08	0.626	0.463	21.08	0.719	0.379	23.06	0.795	0.510
INGP-Base	25.30	0.671	0.371	21.72	0.723	0.330	23.62	0.797	0.423
INGP-Big	25.59	0.699	0.331	21.92	0.745	0.305	24.96	0.817	0.390
M-NeRF360	**27.69**	0.792	0.237	22.22	0.759	0.257	29.40	0.901	0.245
3DGS	27.21	**0.815**	0.214	23.14	0.841	0.183	29.41	**0.903**	0.243
LC-WSR	27.19	0.804	**0.211**	**23.61**	**0.842**	**0.177**	**29.63**	0.902	**0.229**

Quantitative Results

Experiments

Qualcomm AI Research

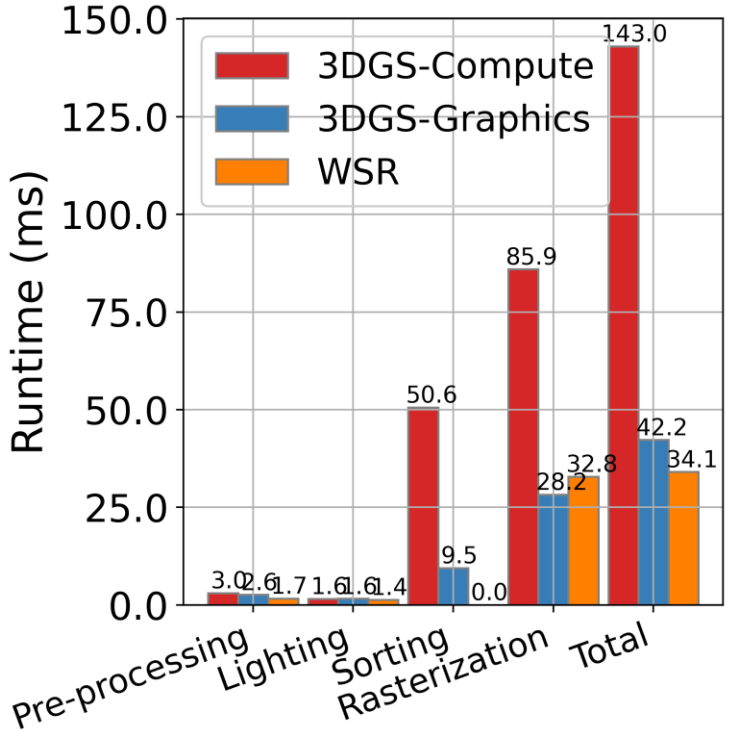
Qualitative Results

Qualcomm AI Research

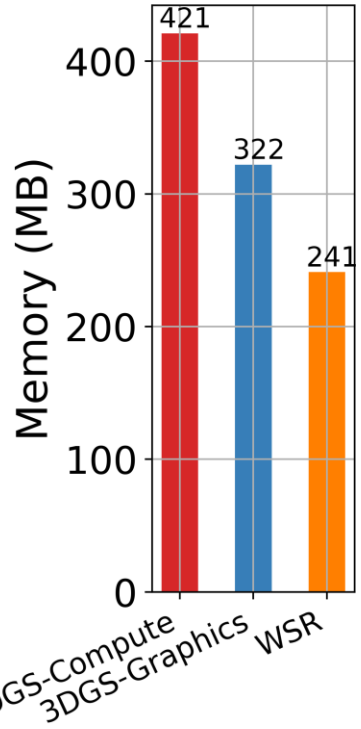
Experiments

3DGS-Compute: Port 3DGS CUDA kernels to Vulkan compute shaders

3DGS-Graphics: Global sorting followed by hardware rasterization



Runtime Comparison



Memory Comparison

On Snapdragon 8 gen 3 at 1920x1080

Pop Artifacts

Moving along z-direction

Thank you!

All the examples were used under a Creative Commons license