Technische Universität Braunschweig
visual computing erlangen
Friedrich-Alexander-Universität Erlangen-Nürnberg

VMV 25
Vision, Modeling, and Visualization

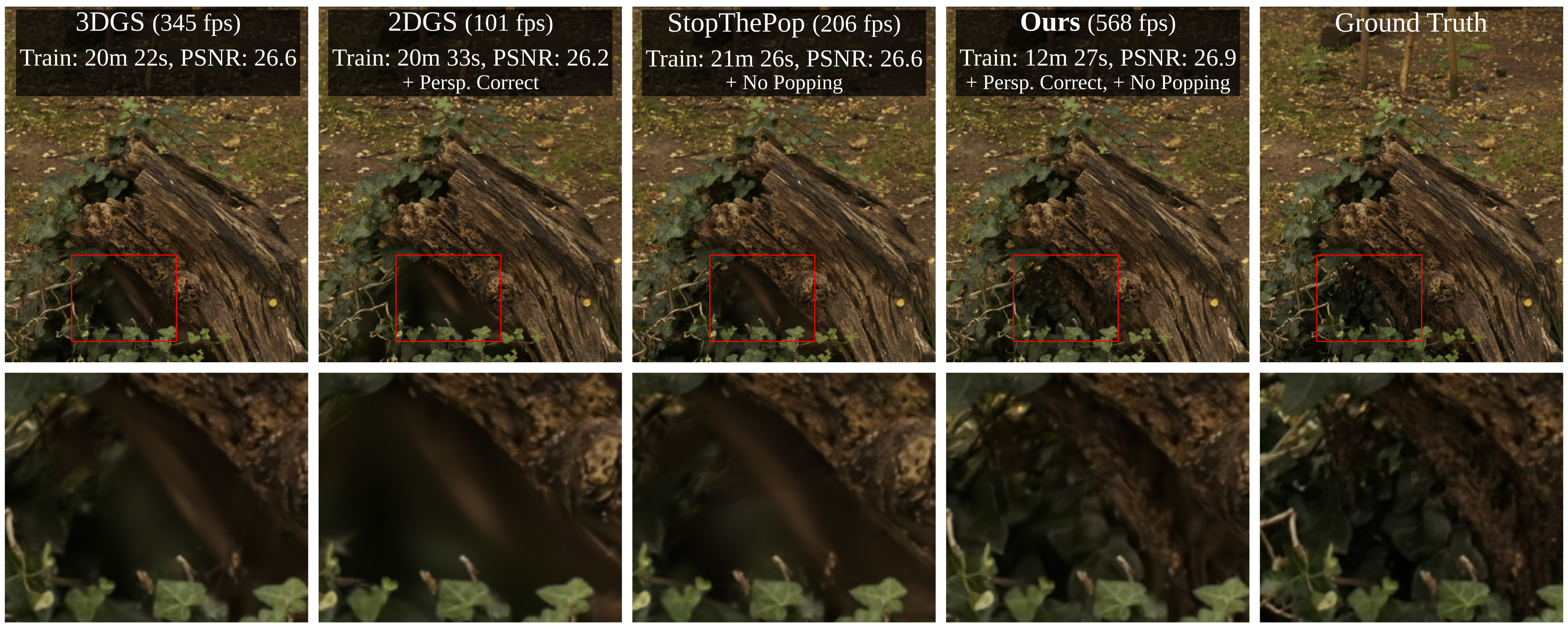
Efficient Perspective-Correct 3D Gaussian Splatting Using Hybrid Transparency

Originally presented at
EUROGRAPHICS LONDON 2025

Florian Hahlbohm Fabian Friederichs Tim Weyrich Linus Franke Moritz Kappel
Susana Castillo Marc Stamminger Martin Eisemann Marcus Magnor

Key Improvements

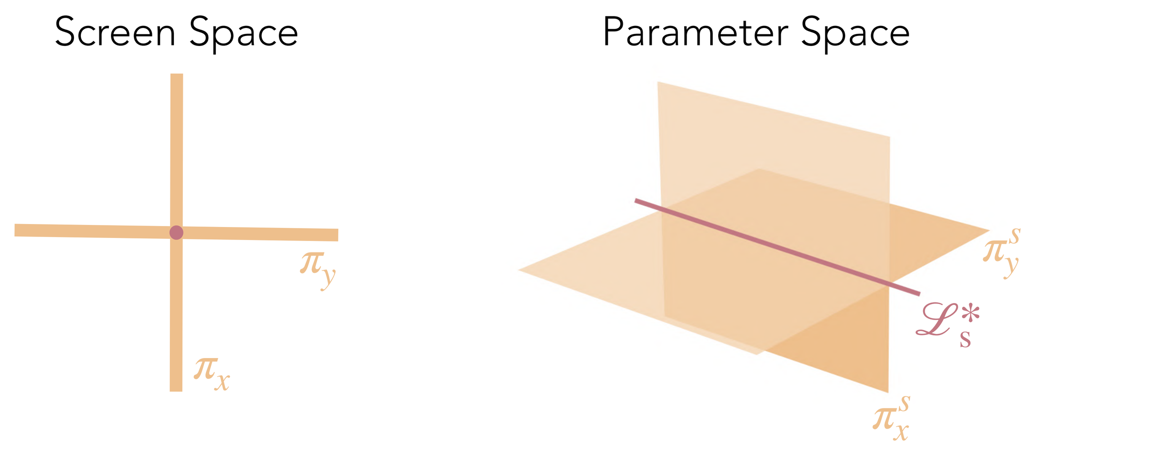
- ✓ Perspective-correct and fast 3D Gaussian rendering
- ✓ Per-pixel sorted blending via hybrid transparency
- ✓ Still works when Gaussians are degenerate
- ✓ Fast training and rendering with equal or better quality



Goal #1: Accurate Splat Bounding and Evaluation

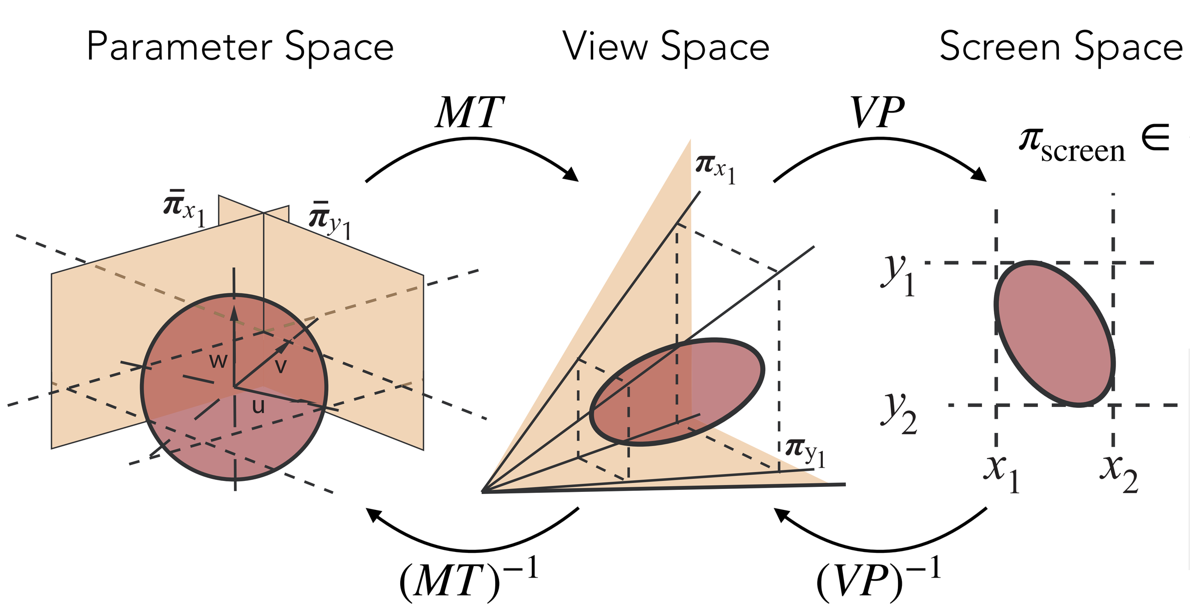
3DGS projects Gaussians onto the screen using an affine approximation of the perspective projection. This causes visually disturbing projection artifacts.

Our proposed solution is a fast and perspectively accurate ray casting approach. To evaluate a Gaussian at a pixel, we represent the corresponding ray as the intersection of two planes and utilize Plücker coordinates to compute the Gaussian's value at the point of maximum contribution along a ray.



$$\pi_x = (1,0,0,-x)^\top$$
$$\pi_y = (0,1,0,-y)^\top$$
$$\pi^s_{x/y} = ((VPMT)^{-1})^{-\top}\pi_{x/y}$$
$$= (VPMT)^\top \pi_{x/y}$$

Transforming planes sidesteps matrix inversion. This makes our approach numerically stable and capable of handling degenerate Gaussians during training.



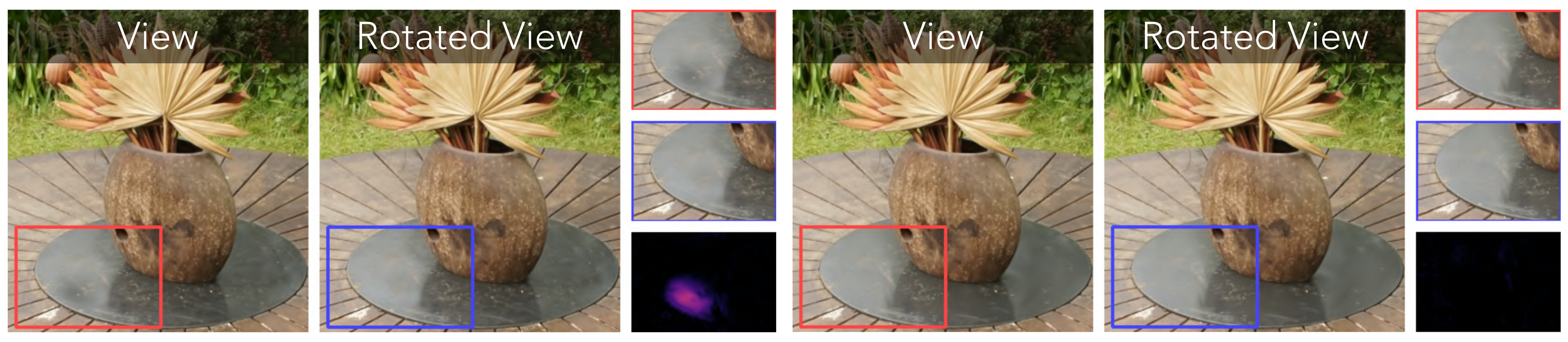
$$\pi_{\text{screen}} \in \{(1,0,0,-x_1), (-1,0,0,x_2), (0,1,0,-y_1), (0,-1,0,y_2), (0,0,1,-z_1), (0,0,-1,z_2)\}$$

T = Gaussian Transform
M = Modelview Transform
P = Projective Transform
V = Viewport Transform

To accelerate rendering, we compute exact screen-space bounding boxes by fitting axis-aligned planes.

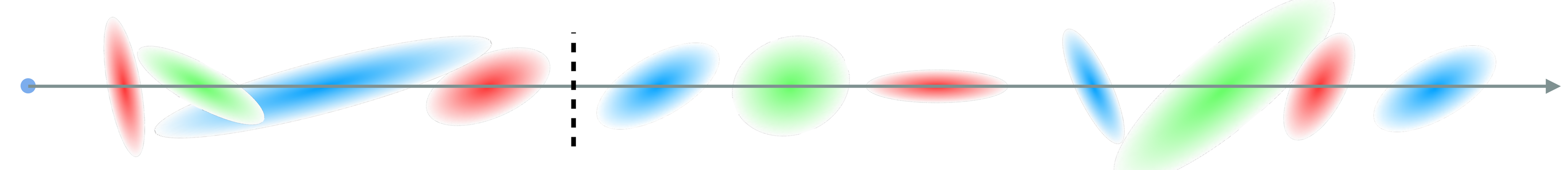
Goal #2: Fast and Temporally-Stable Rendering

3DGS sorts splats based on their mean in view space to approximate prohibitively slow per-pixel sorting. The result are popping artifacts during view rotation.



Global Sorting + Alpha Blending (3DGS) Hybrid Transparency (Ours)

Our hybrid transparency formulation ensures temporal stability with similar quality as fully resolved per-pixel transparencies at a fraction of the rendering costs.



$$C_n = \underbrace{\sum_{i=1}^{K} \alpha_i \, \mathbf{c}_i \prod_{j=1}^{i-1}(1-\alpha_j)}_{\text{Core}} + \underbrace{\prod_{i=1}^{K}(1-\alpha_i)\left((1-\prod_{i=K+1}^{M_n}(1-\alpha_i))\,\frac{\sum_{i=K+1}^{M_n}\alpha_i\,\mathbf{c}_i}{\sum_{i=K+1}^{M_n}\alpha_i}\right)}_{\text{Tail}}$$

To compute the color of a pixel, we alpha blend the K=16 foremost fragments (Core) and use a weighted average to accumulate the remaining ones (Tail).

K=0 K=8 Ours Ours Render w/o Tail Train w/o Tail

Quantitative Comparisons

Method	Mip-NeRF360 [1] (Img. Res. ≈ 1MP–1.5MP) SSIM↑	PSNR↑	LPIPS↓	Training	Render	#Splats	Tanks&Temples [6] (Img. Res. ≈ 2MP) SSIM↑	PSNR↑	LPIPS↓	Training	Render	#Splats
Zip-NeRF [2]	0.828	28.56	0.219	5h	5s	N/a	0.878	26.75	0.233	5h	10s	N/a
3DGS [5]	0.814	27.20	0.254	18m40s	3.2ms	3.31M	0.866	25.27	0.276	20m18s	4.6ms	1.50M
StopThePop [7]	0.814	27.30	0.252	20m11s	4.7ms	3.28M	0.866	24.99	0.270	22m03s	4.8ms	1.49M
↳ w/ opacity decay	0.809	27.08	0.267	12m02s	2.5ms	1.71M	0.862	24.98	0.284	14m22s	3.0ms	0.81M
2DGS [3]	0.795	26.81	0.297	21m08s	7.4ms	2.01M	0.851	24.55	0.314	28m45s	10.4ms	0.95M
Huang et al. [4]	0.814	27.41	0.257	1h01m	10.7ms	3.30M	0.866	25.19	0.276	1h13m	15.7ms	1.49M
GOF [8]	0.821	27.27	0.238	1h21m	60.0ms	2.91M	0.866	25.01	0.274	1h35m	81.3ms	1.26M
Ours	0.822	27.17	0.234	10m07s	2.2ms	2.19M	0.866	24.62	0.272	8m33s	2.6ms	0.81M

Per-Frame Timings

Method	Preprocess	Tiling	Blending	Total	#Splats
	Mip-NeRF360 [1] – Bicycle				
3DGS [5]	1.099ms	3.494ms	3.422ms	8.015ms	6.03M
StopThePop [7]	1.364ms	1.358ms	5.116ms	7.838ms	6.05M
↳ w/ opacity decay	0.554ms	0.547ms	2.442ms	3.543ms	2.91M
2DGS [3]	0.674ms	1.909ms	5.267ms	7.850ms	3.99M
Ours	0.556ms	0.754ms	2.007ms	3.317ms	4.35M
	Tanks&Temples [6] – Francis				
3DGS [5]	0.235ms	2.078ms	1.644ms	3.957ms	0.81M
StopThePop [7]	0.268ms	0.423ms	2.869ms	3.560ms	0.82M
↳ w/ opacity decay	0.154ms	0.310ms	1.980ms	2.444ms	0.45M
2DGS [3]	0.177ms	4.397ms	8.700ms	13.27ms	0.51M
Ours	0.107ms	0.678ms	1.228ms	2.013ms	0.44M

References

[1] Barron et al. Mip-NeRF 360, CVPR 2022.
[2] Barron et al. Zip-NeRF, ICCV 2023.
[3] Huang et al. 2DGS, SIGGRAPH 2024.
[4] Huang et al. 3DGS Error Analysis, ECCV 2024.
[5] Kerbl et al. 3D Gaussian Splatting, TOG 2023.
[6] Knapitsch et al. Tanks and Temples, TOG 2017.
[7] Radl et al. StopThePop, TOG 2024.
[8] Yu et al. GOF, TOG 2024.

Funded by
DFG Deutsche Forschungsgemeinschaft
German Research Foundation
L3S