7DGS: Unified Spatial-Temporal-Angular Gaussian Splatting

Zhongpai Gao*, Benjamin Planche, Meng Zheng, Anwesa Choudhuri, Terrence Chen, Ziyan Wu
United Imaging Intelligence, Boston MA, USA
(zhongpai.gao, benjamin.planche, meng.zheng, anwesa.choudhuri, terrence.chen, ziyan.wu)@uii-ai.com

ICCV OCT 19-23, 2025 HONOLULU HAWAII

Project Link | UII Intern Job

Motivations

- Real-time, photorealistic rendering of **dynamic scenes** with view-dependent effects is still hard.
- 3DGS is fast but limited to static scenes; it can't jointly capture space, time, and viewing direction.
- 4DGS (space+time) and 6DGS (space+direction) solve parts in isolation—there's a need for a unified representation that models their interdependencies without losing real-time speed.

Contributions

- **7DGS Representation**: Over position (3D), time (1D), and direction (3D) with cross-covariances; plus an adaptive refinement module to handle non-rigid motion and time-varying appearance.
- **Conditional Slicing**: Turns a 7D Gaussian into time- and view-conditioned 3D Gaussians, fully compatible with the 3DGS pipeline for efficient real-time rendering.
- **Empirical Validation**: Outperforms prior work (e.g., 4DGS) by up to +7.36 dB PSNR, maintains real-time FPS (≈400+), and uses far fewer Gaussians **across multiple datasets.**

Algorithm 1 Slice 7DGS to Conditional 3DGS

Input: Lower-triangular L, μ_p, μ_t, μ_d, base opacity α, scaling factors λ_t, λ_d, view direction d, observed time t
Output: Conditional μ_{cond}, Σ_{cond}, α_{cond} (optionally, scale S and rotation R are required for densification steps)

1: Reconstruct refined covariance: $\Sigma = LL^\top$.
2: Partition Σ into blocks:

$$\Sigma = \begin{pmatrix} \Sigma_p & \Sigma_{p,(t,d)} \\ \Sigma_{p,(t,d)}^\top & \Sigma_{(t,d)} \end{pmatrix}, \text{with } \Sigma_{(t,d)} = \begin{pmatrix} \Sigma_t & \Sigma_{td} \\ \Sigma_{td}^\top & \Sigma_d \end{pmatrix}$$

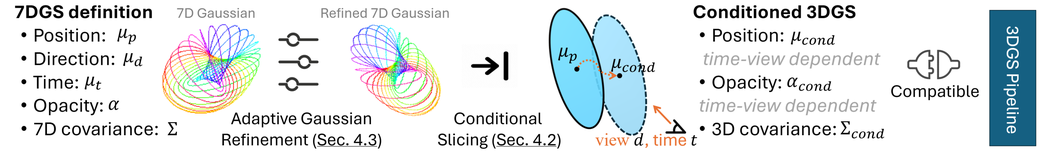
3: Compute conditional statistics:

$$\Sigma_{\text{cond}} = \Sigma_p - \Sigma_{p,(t,d)}\,\Sigma_{(t,d)}^{-1}\,\Sigma_{p,(t,d)}^\top,$$
$$\mu_{\text{cond}} = \mu_p + \Sigma_{p,(t,d)}\,\Sigma_{(t,d)}^{-1}\begin{pmatrix} t-\mu_t \\ d-\mu_d \end{pmatrix}.$$

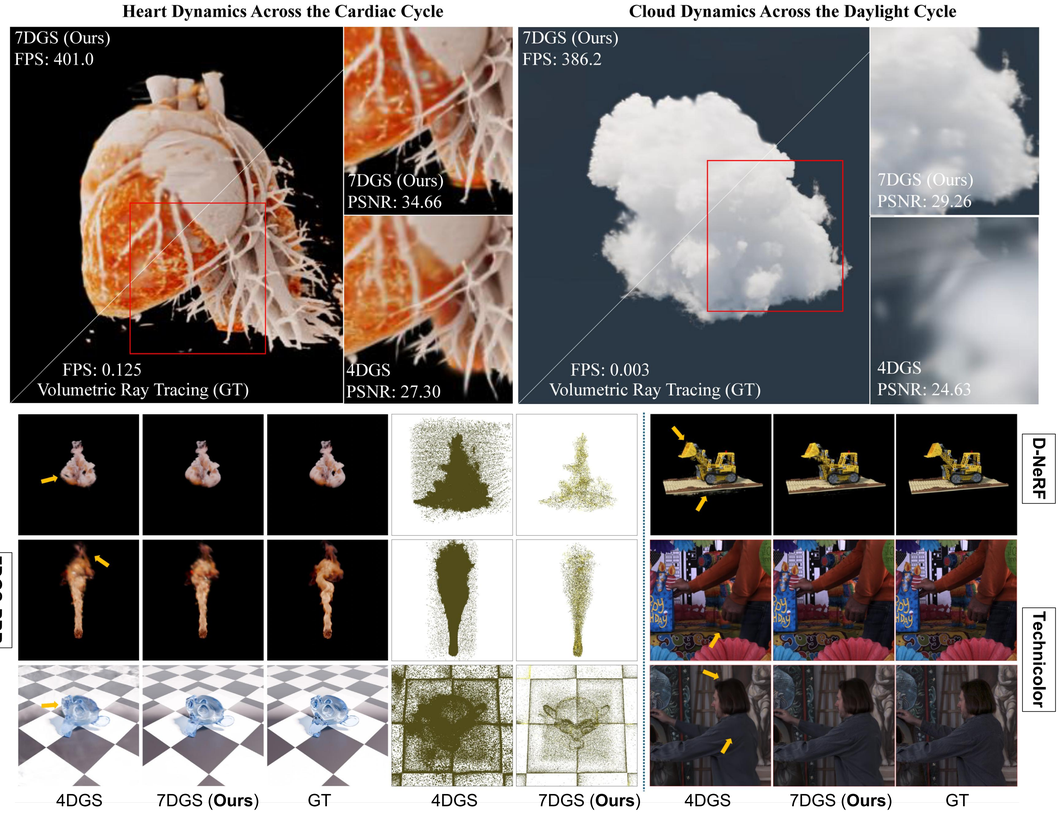
4: Compute conditional opacity:

$$f_{\text{temp}} = \exp\!\left(-\tfrac{1}{2}\,\lambda_t\,(t-\mu_t)^2\,\Sigma_t^{-1}\right),$$
$$f_{\text{dir}} = \exp\!\left(-\tfrac{1}{2}\,\lambda_d\,(d-\mu_d)^\top\Sigma_d^{-1}(d-\mu_d)\right),$$
$$\alpha_{\text{cond}} = \alpha \cdot f_{\text{temp}} \cdot f_{\text{dir}}$$

Our 7DGS Compatible with the Existing 3DGS Pipeline



Quantitative Results



Quantitative Results

Dataset	Scene	4DGS PSNR↑	4DGS SSIM↑	4DGS LPIPS↓	4DGS Train↓	4DGS FPS↑	4DGS # points↓	7DGS (Ours) PSNR↑	7DGS (Ours) SSIM	7DGS (Ours) LPIPS↓	7DGS (Ours) Train↓	7DGS (Ours) FPS↑	7DGS (Ours) # points↓
7DGS-PBR	heart1	27.30	0.949	0.046	**103.0**	186.9	694,006	**35.48**	**0.986**	**0.020**	114.2	155.7	82,813
	heart2	25.13	0.920	0.084	**103.4**	160.4	869,245	**31.80**	**0.964**	**0.051**	145.9	139.5	98,458
	cloud	24.63	0.938	0.100	123.7	219.0	216,878	**29.60**	**0.955**	**0.075**	**102.6**	199.2	44,858
	dust	35.88	0.954	0.037	97.0	296.1	357,744	**37.30**	**0.956**	**0.037**	**69.8**	243.9	11,253
	flame	29.34	0.928	0.067	113.7	151.2	947,786	**32.53**	**0.940**	**0.059**	**74.1**	247.2	16,544
	suzanne	24.45	0.917	0.141	222.5	141.8	766,098	**28.26**	**0.949**	**0.062**	**193.9**	62.5	336,713
	avg	27.79	0.934	0.079	127.2	192.6	641,960	**32.50**	**0.958**	**0.051**	**116.7**	174.7	98,440
D-NeRF	b.balls	33.24	0.982	0.025	**50.7**	219.9	276,073	**35.10**	**0.984**	**0.019**	86.6	104.3	129,791
	h.warrior	**34.10**	**0.949**	**0.067**	35.3	299.4	298,391	32.96	0.935	0.084	**31.3**	238.3	9,569
	hook	**32.93**	**0.970**	**0.034**	38.0	325.1	174,720	31.57	0.962	0.040	**35.7**	233.1	24,700
	j.jacks	31.14	0.970	0.044	66.9	366.0	143,665	**33.57**	**0.977**	**0.027**	**34.1**	243.2	18,784
	lego	25.58	0.917	0.077	**55.4**	320.0	186,165	**28.86**	**0.947**	**0.051**	78.5	160.2	74,884
	mutant	39.01	0.991	0.009	**39.1**	341.6	138,691	**41.36**	**0.995**	**0.005**	42.5	193.7	37,706
	standup	39.75	0.991	0.008	34.4	330.4	142,468	**40.60**	**0.992**	**0.008**	**33.5**	224.0	15,598
	trex	29.89	0.979	0.021	100.7	169.2	682,378	**30.72**	**0.980**	**0.018**	**63.4**	156.5	67,994
	avg	33.21	0.969	0.036	52.6	296.4	255,319	**34.34**	**0.972**	**0.032**	**50.7**	194.2	47,378
Technicolor	birthday	31.28	0.922	0.153	370.8	69.6	842,491	**32.31**	**0.940**	**0.111**	**117.0**	39.2	589,128
	fabien	**35.48**	**0.894**	**0.297**	332.8	110.5	705,106	34.87	0.885	0.317	**73.0**	131.3	107,240
	painter	35.09	0.905	0.238	361.9	99.4	287,036	**36.54**	**0.919**	**0.208**	**98.1**	113.1	144,226
	theater	**31.84**	0.871	0.291	383.7	83.0	946,909	31.54	**0.876**	**0.265**	**87.3**	94.0	197,713
	train	32.58	0.932	0.102	345.3	58.3	1,412,917	**32.64**	**0.940**	**0.089**	**185.2**	18.6	1,043,645
	avg	33.25	0.905	0.216	358.9	84.2	838,892	**33.58**	**0.912**	**0.198**	**112.1**	79.2	416,390

SOTA Comparison

	Method	PSNR↑	SSIM↑	LPIPS↓
D-NeRF	D-NeRF [27]	29.67	0.95	0.07
	HexPlane [2]	31.05	0.97	0.04
	K-Planes [5]	31.61	0.97	-
	DaReNeRF [19]	31.95	0.97	**0.03**
	4DGS* [38]	33.21	0.97	0.04
	4DGaussians [35]	33.30	**0.98**	**0.03**
	7DGS (**Ours**)	**34.34**	0.97	**0.03**

	Method	PSNR↑	SSIM↑	LPIPS-Alex↓
Technicolor	DyNeRF [15]	31.80	-	0.142
	HyperReel [26]	32.73	0.906	0.109
	4DGaussians [35]	30.79	0.843	0.178
	STG [16]	33.23	0.912	**0.085**
	4DGS* [38]	33.25	0.905	0.110
	Ex4DGS* [14]	33.49	**0.917**	0.094
	7DGS (**Ours**)	**33.58**	0.912	0.101

Ablation

Dataset	Method	PSNR	SSIM	LPIPS	Train	# points	MB
7DGS-PBR	4DGS	27.79	0.934	0.079	127.2	641,960	397.6
	7DGS w/o direction	30.83	0.952	0.060	**115.3**	103,618	34.8
	7DGS $\lambda_t = \lambda_d = 0.8$	32.41	**0.958**	**0.051**	116.1	**94,999**	**31.9**
	7DGS (**Ours**)	**32.50**	**0.958**	**0.051**	116.7	98,440	33.1
N3V	4DGS (sparse input)	30.75	0.938	0.063	433.9	471,944	219.4
	7DGS (sparse input)	**31.57**	**0.947**	**0.052**	**172.3**	**245,553**	**107.5**

Summary

We introduce **7DGS**, a unified spatial–temporal–angular Gaussian splatting method that extends 3DGS by adding time (1D) and viewing direction (3D) to the spatial parameters. A single set of Gaussians jointly models non-rigid motion and view-dependent appearance, while a lightweight conditioning / slicing step produces time- and view-conditioned 3D Gaussians that render with standard 3DGS pipelines in real time. This yields temporally consistent reconstructions of dynamic scenes with sharper specular and reflective effects, all while remaining plug-and-play compatible with existing 3DGS toolchains and optimizers.