Real-time Photorealistic Dynamic Scene Representation and Rendering with 4D Gaussian Splatting

Zeyu Yang, Hongye Yang, Zijie Pan, Li Zhang
Fudan University

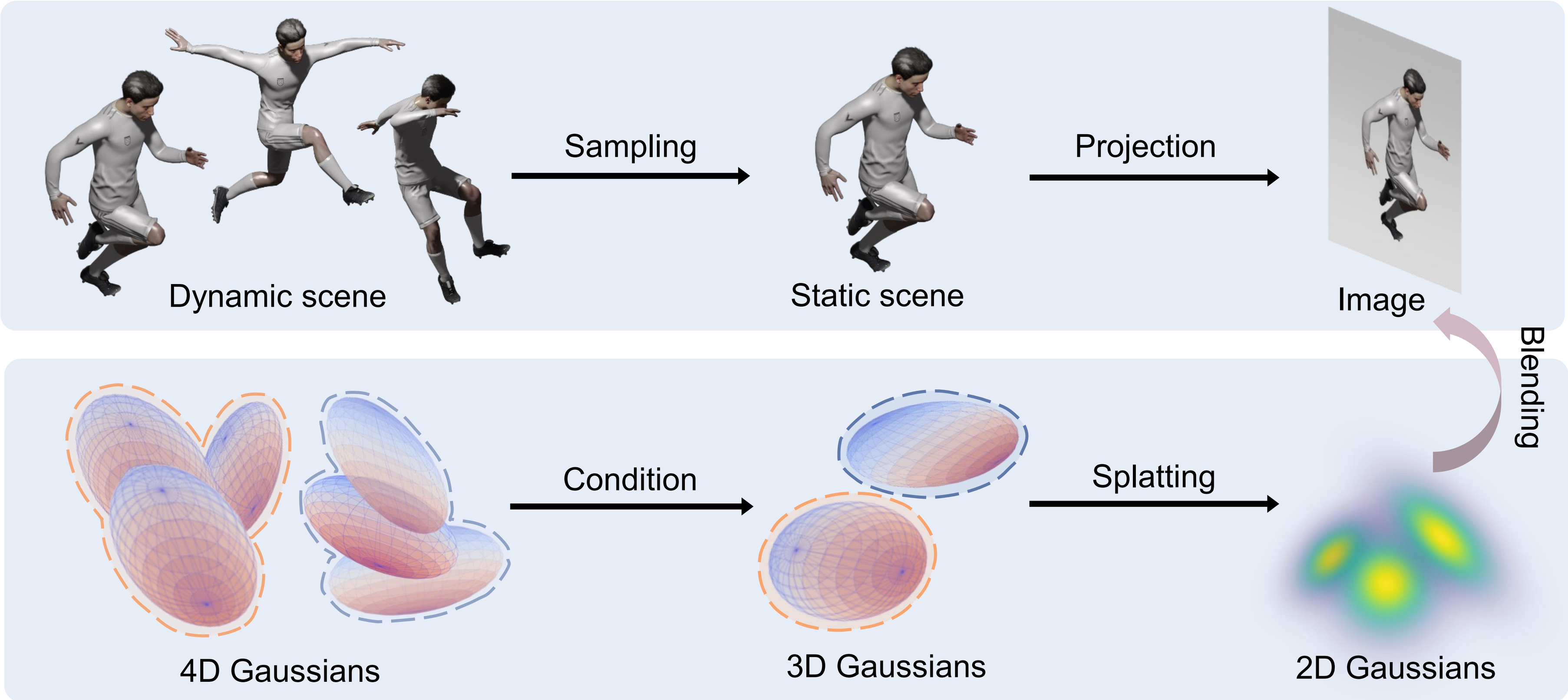
Contact: lizhangfd@fudan.edu.cn

Paper, code, and demo are available:
https://github.com/fudan-zvg/4d-gaussian-splatting

4D Gaussian Splatting

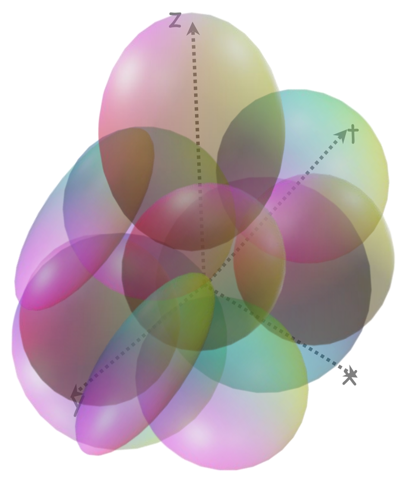
Overview

- The reconstruction of a dynamic scene can be recast as optimizing a series of native 4D Gaussians to fit its underlying spatiotemporal 4D volume.
- Efficient end-to-end optimization.
- Fully interpretable representation, which inherits the merits of 3D Gaussian Splatting, thus friendly for editing and composition.



- The rendering process of 4D Gaussian Splatting has conceptual parallels with the imaging process of a dynamic scenes:
  - 4D Gaussian ⟷ dynamic scene
  - Conditional 3D Gaussian ⟷ static scene in given time
  - Planar Gaussian ⟷ 2D image

Parameterization of 4D Gaussian



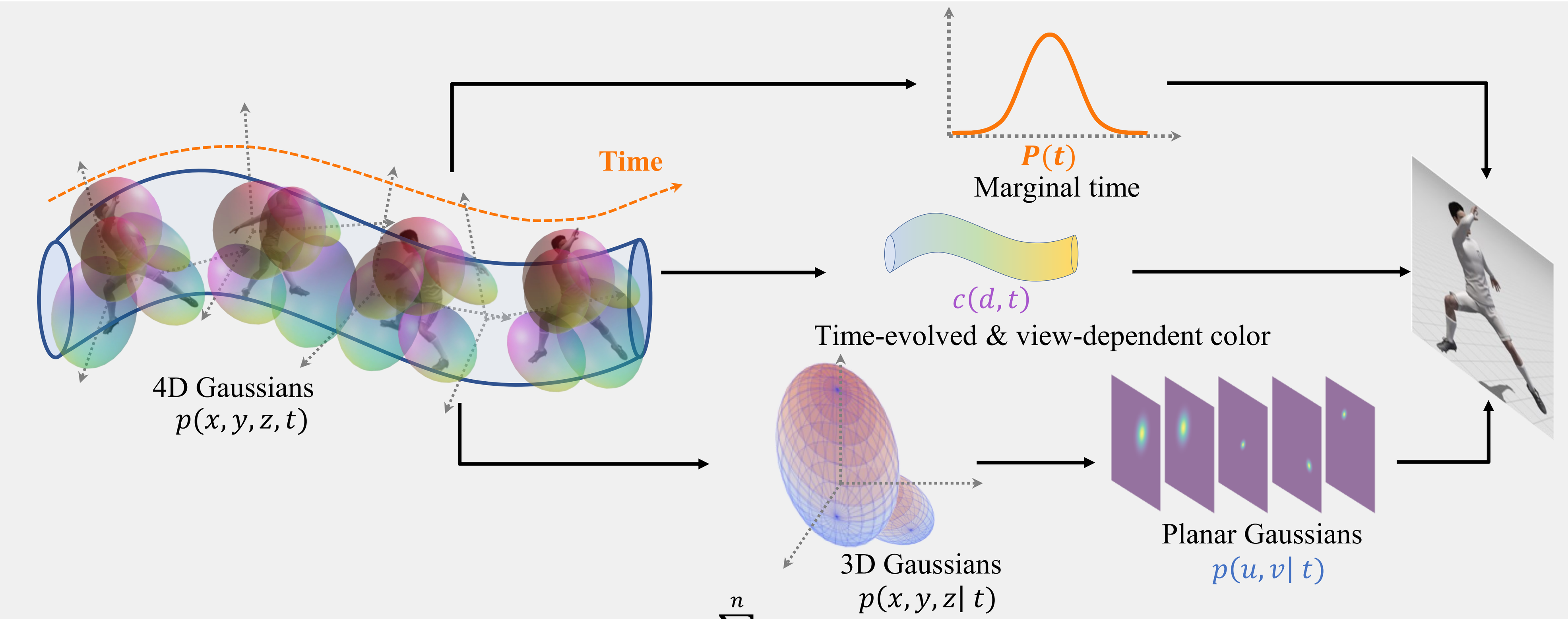
- $p = \begin{bmatrix} p_x \\ p_y \\ p_z \\ p_t \end{bmatrix}$ opacity SH

- $\Sigma = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} & \sigma_{xt} \\ & \sigma_{yy} & \sigma_{yz} & \sigma_{yt} \\ & & \sigma_{zz} & \sigma_{zt} \\ & & & \sigma_{tt} \end{bmatrix} = RSS^TR^T$ $\quad S = \mathrm{diag}(s_x, s_y, s_z, s_t)$

- $R = L(q_l)R(q_r) = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} \begin{bmatrix} p & -q & -r & -s \\ q & p & s & -r \\ r & -s & p & q \\ s & r & -q & p \end{bmatrix}$ $\quad q_l = (a, b, c, d)$ $\quad q_r = (p, q, r, s)$

Efficient rendering via 4D Gaussian splatting

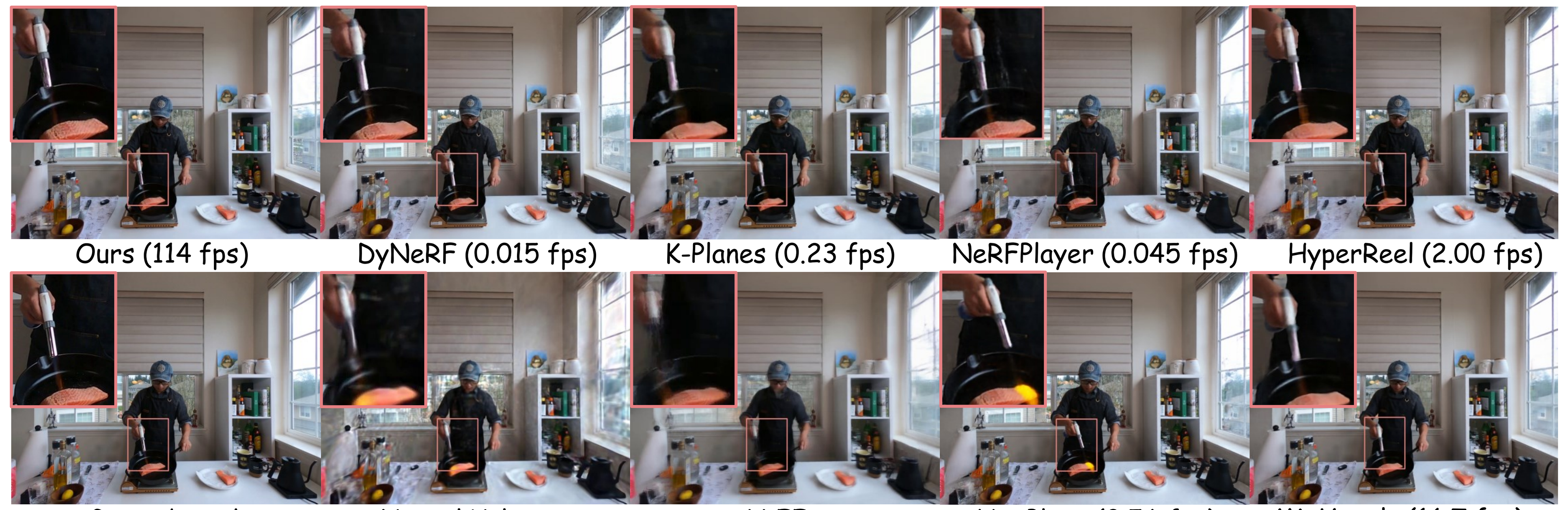
- The influence of a 4D Gaussian on a pixel at time t can be evaluated by first decomposing it into its spatial conditional distribution and temporal marginal distribution, which are 3D and 1D Gaussians respectively, and then splatting the conditional 3D Gaussian weighted by the marginal Gaussian at given viewpoint.



$$\mathcal{I}(u,v,t) = \sum_{i=1}^{n} \alpha_i p_i(u,v|t) p_i(t) c(d_i,t) T_i$$

Results

- State-of-the-art performance and efficiency for neural volumetric video.



Ours (114 fps) DyNeRF (0.015 fps) K-Planes (0.23 fps) NeRFPlayer (0.045 fps) HyperReel (2.00 fps)

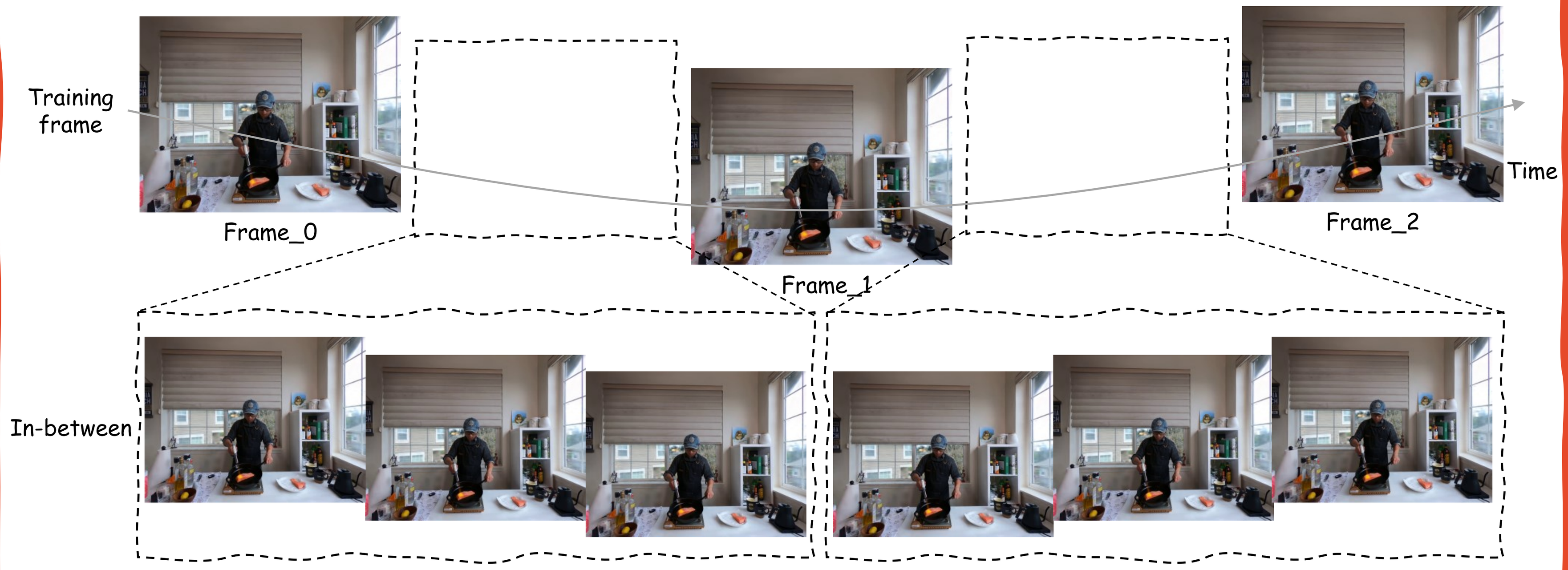
Ground truth Neural Volumes LLFF HexPlane (0.56 fps) MixVoxels (16.7 fps)

Method	PSNR ↑	DSSIM ↓	LPIPS ↓	FPS ↑
- Plenoptic Video (real, multi-view)				
Neural Volumes (Lombardi et al., 2019)[1]	22.80	0.062	0.295	-
LLFF (Mildenhall et al., 2019)[1]	23.24	0.076	0.235	-
DyNeRF (Li et al., 2022b)[1]	29.58	0.020	0.099	0.015
HexPlane (Cao & Johnson, 2023)	31.70	0.014	0.075	0.56[3]
K-Planes-explicit (Fridovich-Keil et al., 2023)	30.88	0.020	-	0.23[3]
K-Planes-hybrid (Fridovich-Keil et al., 2023)	31.63	0.018	-	-
MixVoxels-L (Wang et al., 2023)	30.80	0.020	0.126	16.7
StreamRF (Li et al., 2022a)[1]	29.58	-	-	8.3
NeRFPlayer (Song et al., 2023)	30.69	0.035[2]	0.111	0.045
HyperReel (Attal et al., 2023)	31.10	0.037[2]	0.096	2.00
4DGS (Wu et al., 2023)[4]	31.02	0.030	0.150	36
4DGS (Ours)	**32.01**	**0.014**	**0.055**	**114**

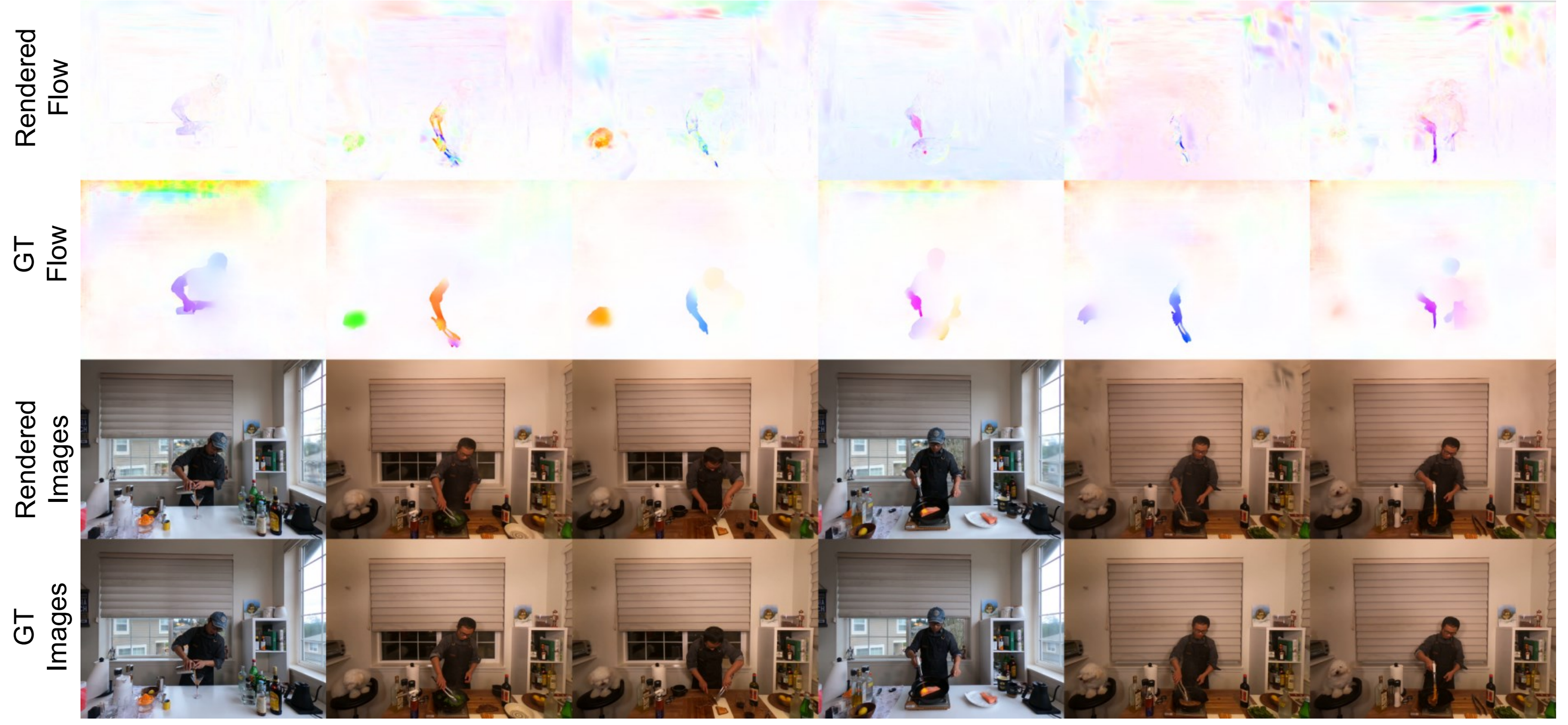
Method	PSNR ↑	SSIM ↑	LPIPS ↓
- D-NeRF (synthetic, monocular)			
T-NeRF (Pumarola et al., 2021)	29.51	0.95	0.08
D-NeRF (Pumarola et al., 2021)	29.67	0.95	0.07
TiNeuVox (Fang et al., 2022)	32.67	0.97	0.04
HexPlanes (Cao & Johnson, 2023)	31.04	0.97	0.04
K-Planes-explicit (Fridovich-Keil et al., 2023)	31.05	0.97	-
K-Planes-hybrid (Fridovich-Keil et al., 2023)	31.61	0.97	-
V4D (Gan et al., 2023)	33.72	0.98	0.02
4DGS (Wu et al., 2023)[1]	33.30	0.98	0.03
4DGS (Ours)	**34.09**	**0.98**	**0.02**

Results

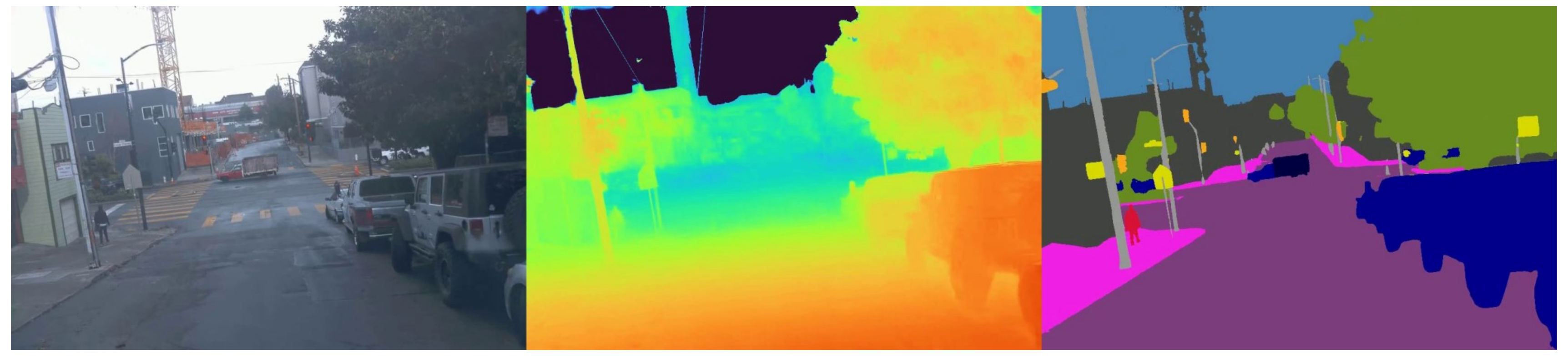
- 4D Gaussian Splatting can correctly model the smooth inter-frame motions.



- 4D Gaussian Splatting can capture the underlying dynamics with only photometric loss.



- 4D Gaussian Splatting can be also applied for the reconstruction of large-scale urban scenes without relying on the laborious foreground annotations.



* Videos corresponding to all the images above are available at: https://fudan-zvg.github.io/4d-gaussian-splatting.