Modeling Dynamic Scenes with Native 4D Primitives

2025 winter seminar

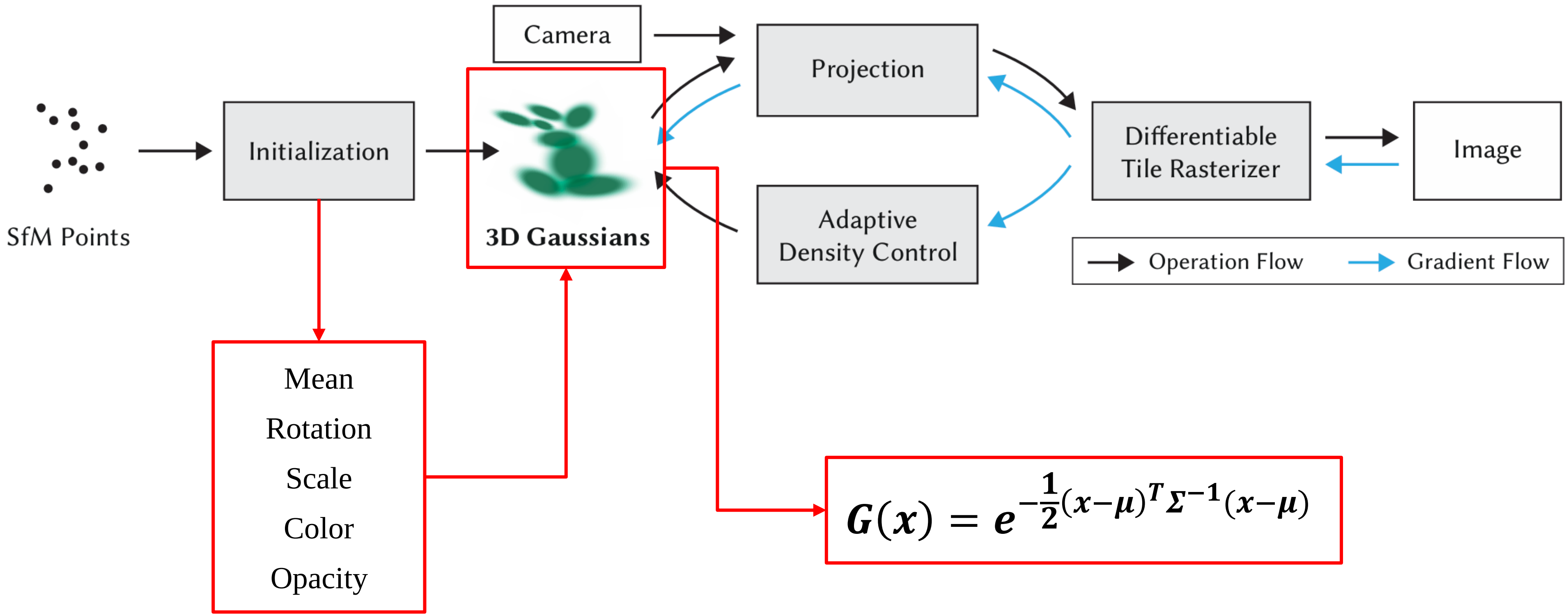
Sogang University
Vision & Display Systems Lab, Dept. of Electronic Engineering

Presented By
Jimin Roh

Real-time Photorealistic Dynamic Scene Representation and Rendering with 4D Gaussian Splatting
ICLR 2024

Background

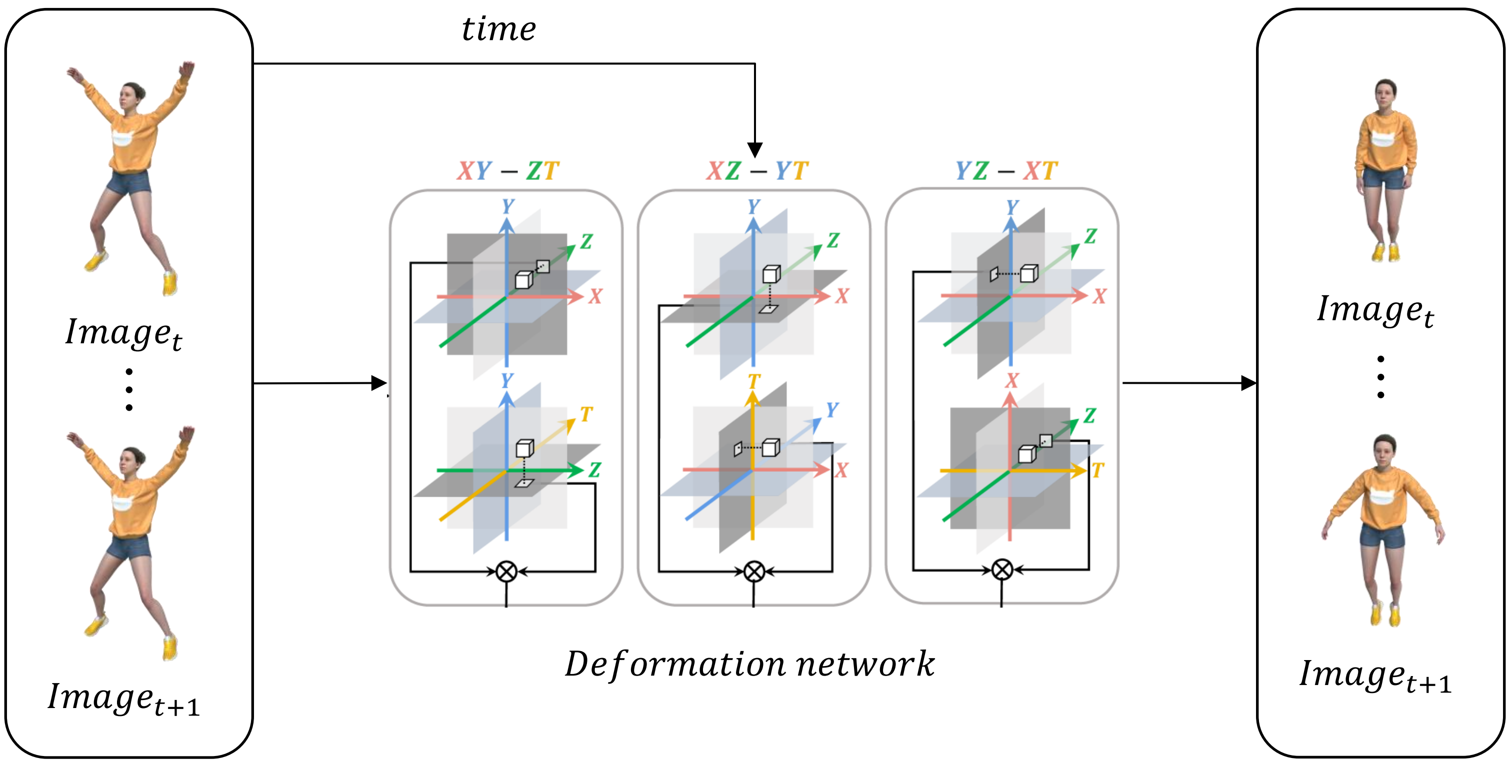
- 3D Gaussian



$$G(x) = e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

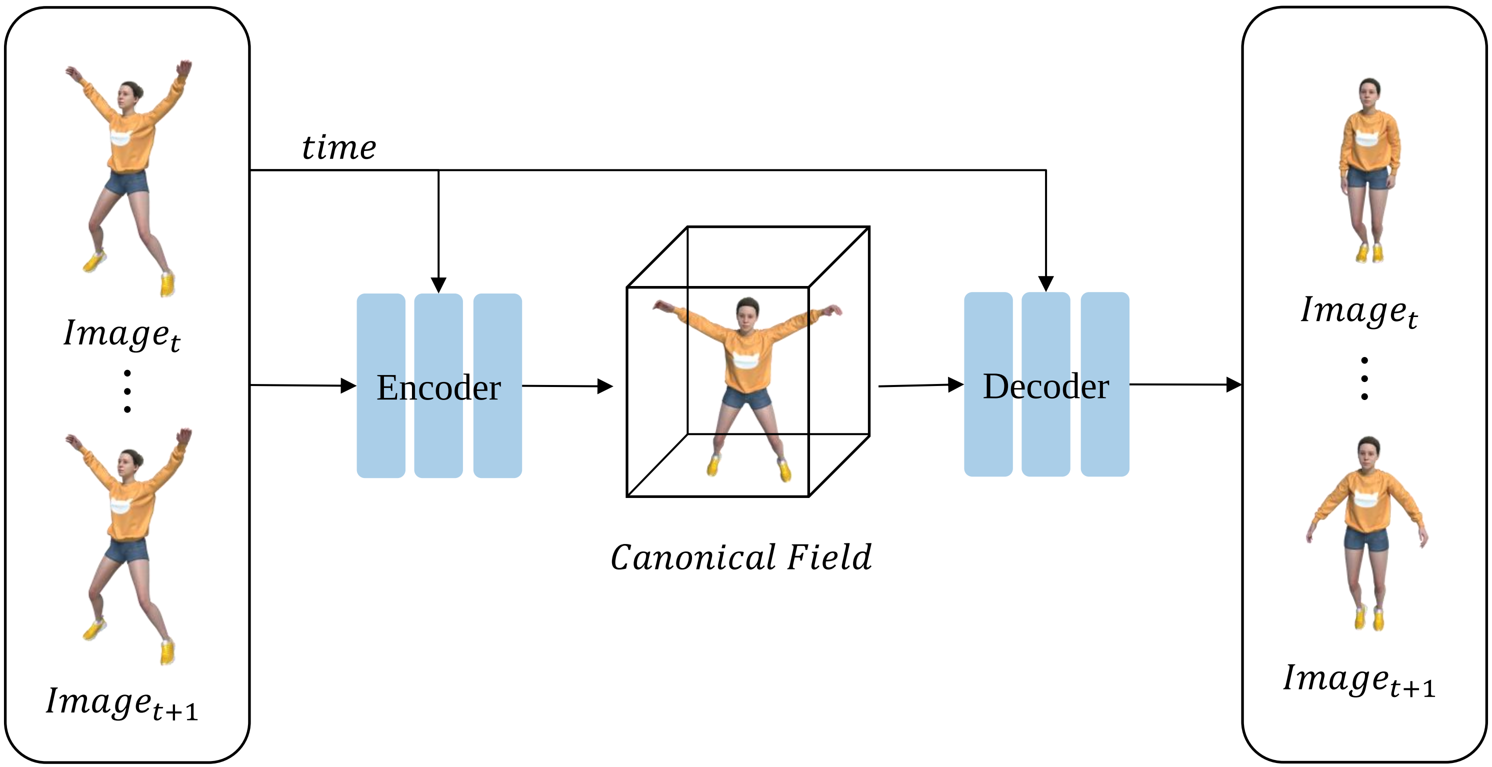
Introduction

- Problem formulation
  - Without canonical field
    - DyNeRF, HexPlane, HyperReel



Introduction

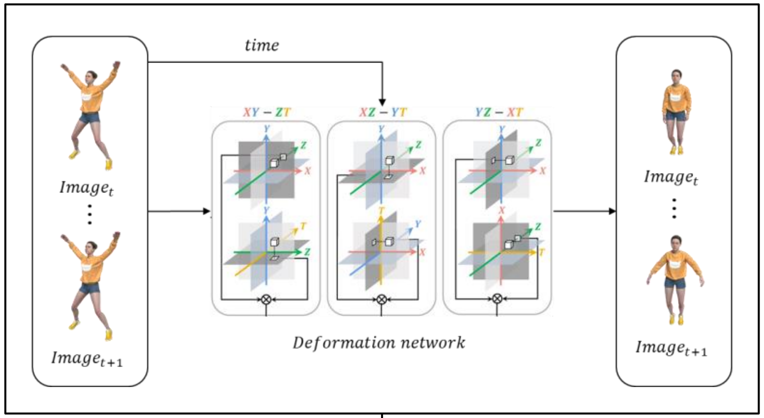
- Problem formulation
  - With canonical field
    - D-NeRF, Nerfplayer, Particlenerf, Dynamic 3DGS



Introduction

- Problem formulation

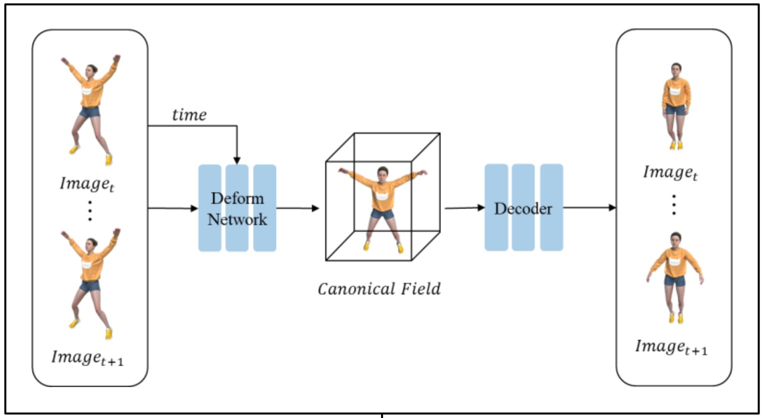
Without canonical field



장면의 모션을 명시적으로
정의하지 않음

- MLP, grid, low-rank decomposition 방식은 주어진 데이터에 의존적
- Spatial-temporal이 독립적으로 학습

With canonical field

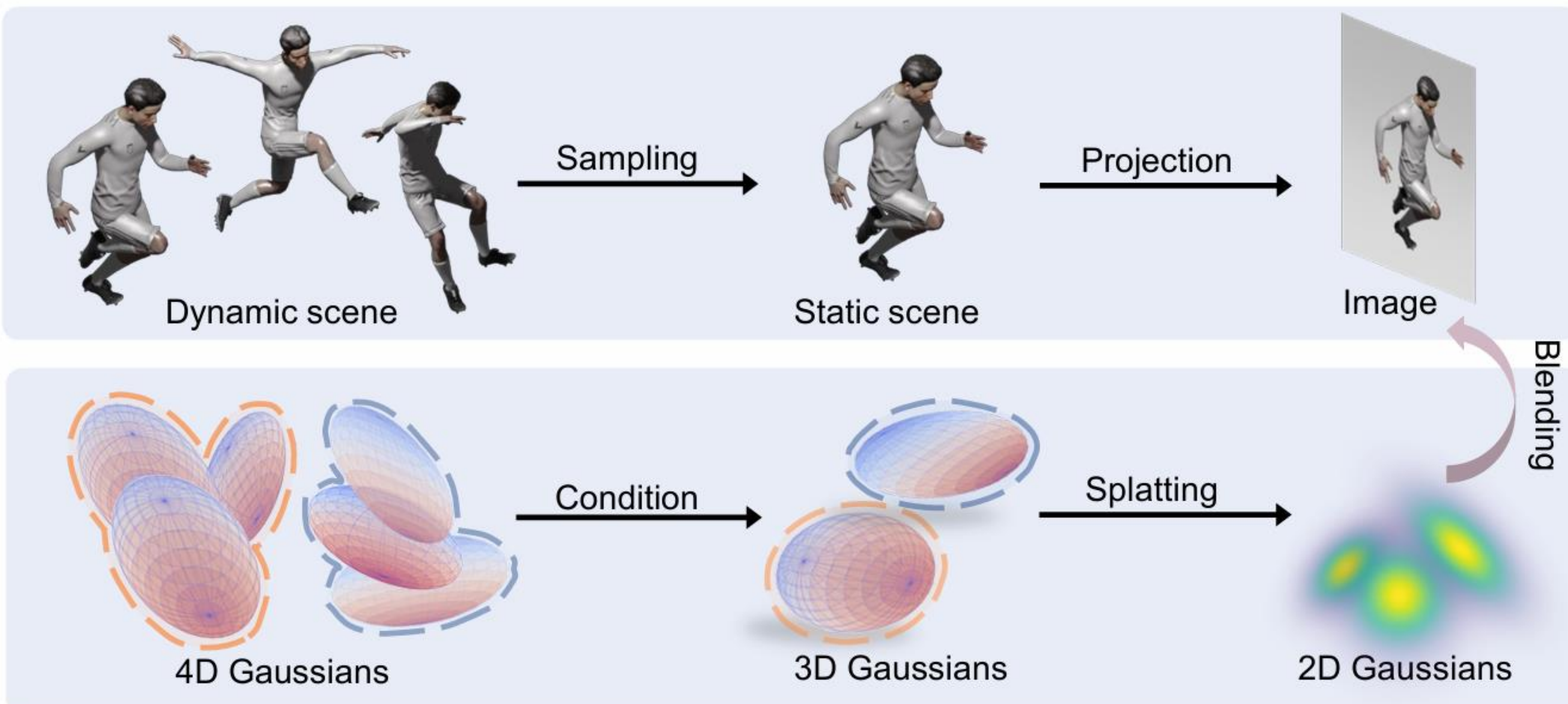


Time을 고려한 space를 정의

- 복잡한 실제 장면에서 유연성과 확장성이 감소

Introduction

- Overall pipeline
    - 직접적으로 4D Gaussian을 모델링

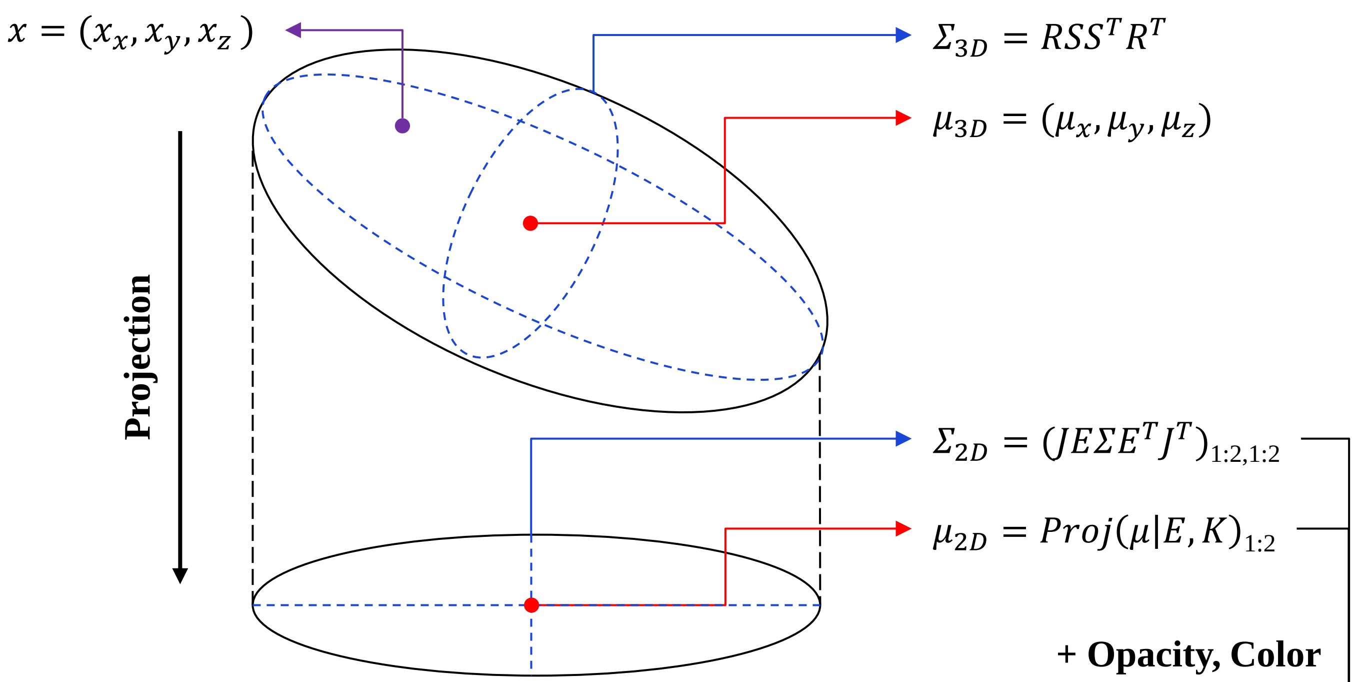


Method 1. Representation of 4D Gaussian

- 3D Gaussian

$$G(x) = e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)}$$

Mean, Rotation, Scale



$$I(u,v) = \sum_{i=1}^{N} p_i(u,v;\mu_{2D},\Sigma_{2D})\alpha_i c_i(d_i) \prod_{j=1}^{i-1}(1 - p_i(u,v;\mu_{2D},\Sigma_{2D})\alpha_i)$$

Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Attribute: Mean, Time

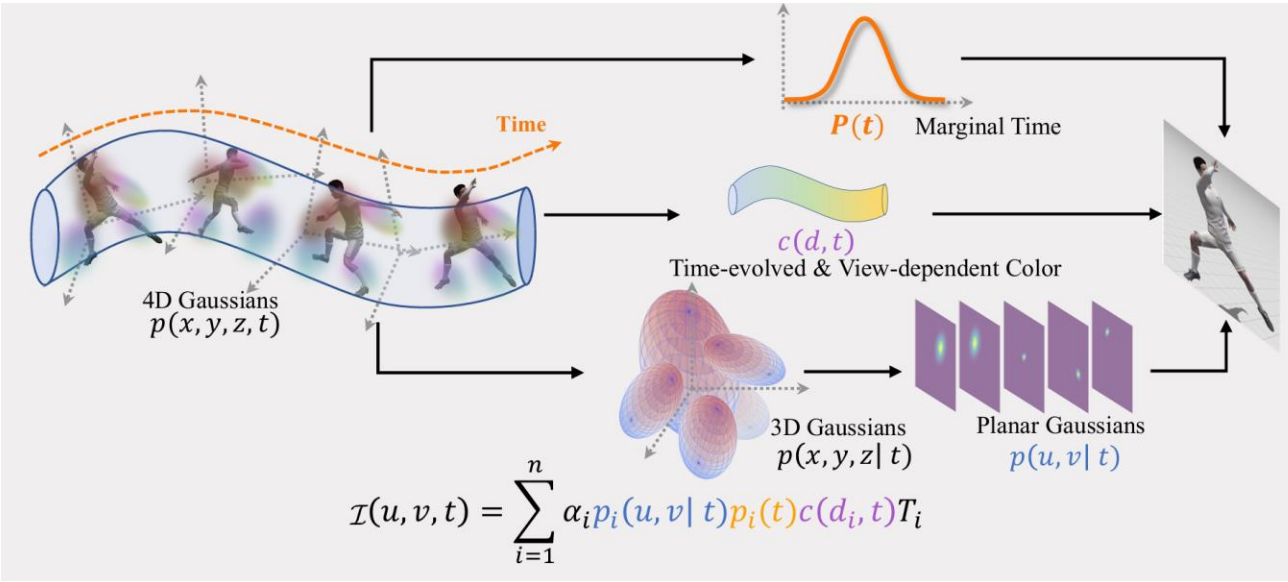
3D

$$I(u,v) = \sum_{i=1}^{N} p_i(u,v)\alpha_i c_i(d_i) \prod_{j=1}^{i-1}(1 - p_i(u,v)\alpha_i)$$

4D

$$I(u,v) = \sum_{i=1}^{N} p_i(u,v,t)\alpha_i c_i(d_i,t) \prod_{j=1}^{i-1}(1 - p_i(u,v,t)\alpha_i)$$

$$I(u,v) = \sum_{i=1}^{N} p_i(t) p_i(u,v|t)\alpha_i c_i(d_i,t) \prod_{j=1}^{i-1}(1 - p_i(t) p_i(u,v|t)\alpha_i)$$



Method 1. Representation of 4D Gaussian

- 4D Gaussian

$$I(u,v) = \sum_{i=1}^{N} p_i(t) p_i(u,v|t) \alpha_i c_i(d_i,t) \prod_{j=1}^{i-1} (1 - p_i(t) p_i(u,v|t) \alpha_i)$$

  - Attribute: Rotation, Scale

$$\Sigma_{3D} = RSS^T R^T$$

(3x3 matrix)

$$q = \begin{pmatrix} r \\ x \\ y \\ z \end{pmatrix} \quad R = \begin{pmatrix} 1-2(y^2+z^2) & 2(xy-rz) & 2(xz+ry) \\ 2(xy+rz) & 1-2(x^2+z^2) & 2(yz-rz) \\ 2(xz-ry) & 2(yz+rx) & 1-2(x^2+y^2) \end{pmatrix}$$

(3x3 matrix)

$$s = (s_x, s_y, s_z) \quad S = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix}$$

(4x4 matrix)

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

(4x4 matrix)

$$s = (s_x, s_y, s_z, s_t) \quad S = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & s_t \end{pmatrix}$$

$$\Sigma_{xyzt} = RSS^T R^T$$

$$\mu_{xyzt} = (\mu_x, \mu_y, \mu_z, \mu_t)$$

4D to 3D →

$$\Sigma_{xyz|t} = \Sigma_{1:3,1:3} - \Sigma_{1:3,4} \Sigma^{-1}_{4,4} \Sigma_{4,1:3}$$

$$\mu_{xyz|t} = \mu_{1:3} + \Sigma_{1:3,4} \Sigma^{-1}_{4,4} (t - \mu_t)$$

서강대학교 SOGANG UNIVERSITY

VDS LAB

Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Attribute: Rotation, Scale

$$I(u,v)=\sum_{i=1}^{N}p_i(t)p_i(u,v|t)\alpha_i c_i(d_i,t)\prod_{j=1}^{i-1}(1-p_i(t)p_i(u,v|t)\alpha_i)$$

4D to 3D

$$\boldsymbol{\Sigma_{xyzt}=RSS^TR^T}$$

$$\boldsymbol{\mu_{xyzt}=(\mu_x,\mu_y,\mu_z,\mu_t)}$$

x,y,z 간의 관계

t의 scale을 조정

$$\boldsymbol{\Sigma_{xyz|t}=\Sigma_{1:3,1:3}-\Sigma_{1:3,4}\Sigma^{-1}_{4,4}\Sigma_{4,1:3}}$$

x,y,z와 t 간의 관계

t에서 x,y,z로 관계

$$\boldsymbol{\mu_{xyz|t}=\mu_{1:3}+\Sigma_{1:3,4}\Sigma^{-1}_{4,4}(t-\mu_t)}$$

결국 4D Gaussian에서 time에 대한 영향을 제거하여 3D Gaussian을 생성하는 과정

서강대학교 SOGANG UNIVERSITY

VDS LAB

Method 2. 4D Spherindrical Harmonics

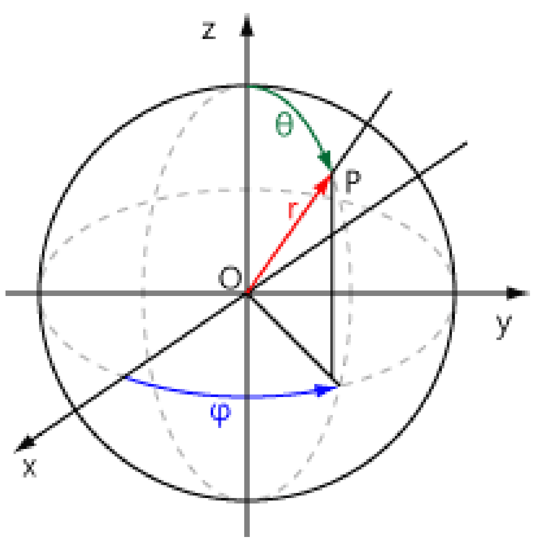
- 4D Gaussian
  - Attribute: Color

$$I(u,v) = \sum_{i=1}^{N} p_i(t)p_i(u,v|t)\alpha_i c_i(d_i,t) \prod_{j=1}^{i-1}(1 - p_i(t)p_i(u,v|t)\alpha_i)$$

Spherical Harmonics Coefficients

$$f(\theta,\varphi) = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} c_l^m Y_l^m(\theta,\varphi)$$

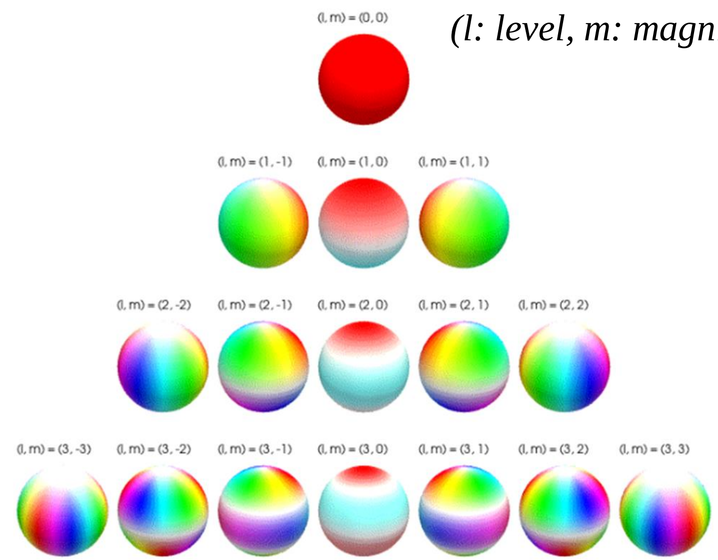
Spherical Harmonics



$$Y_l^m(\theta,\phi) = \sqrt{\frac{(2l+1)(l-|m|)!}{4\pi(l+|m|)!}}\,P_l^{|m|}(\cos\theta)e^{im\phi}$$

$$P_\ell^{(|m|)}(\cos\theta) = (-1)^m\frac{(\ell+|m|)!}{(\ell-|m|)!}P_\ell^{(-|m|)}(\cos\theta)$$

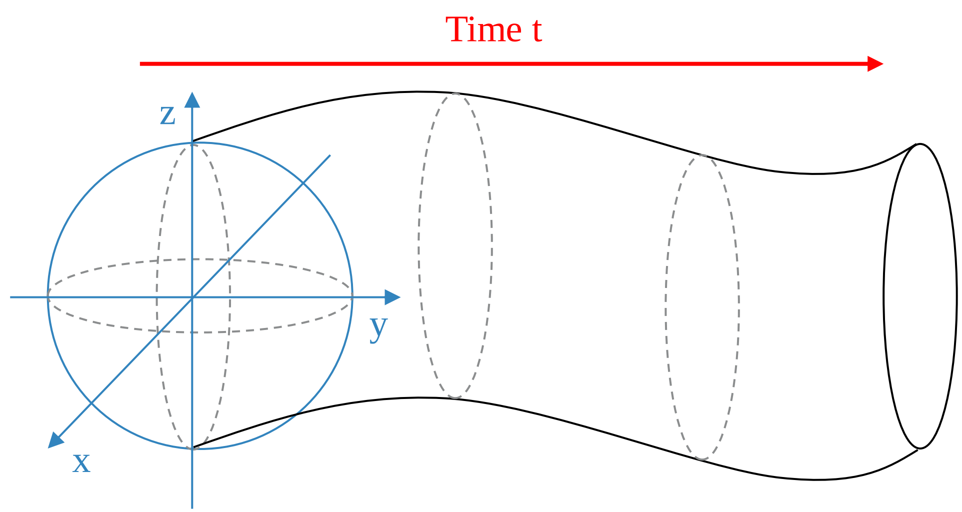
(l: level, m: magnitude)



Method 2. 4D Spherindrical Harmonics

- 4D Gaussian
  - Attribute: Color

$$I(u,v) = \sum_{i=1}^{N} p_i(t)p_i(u,v|t)\alpha_i c_i(d_i,t) \prod_{j=1}^{i-1}(1 - p_i(t)p_i(u,v|t)\alpha_i)$$



$$Z_{nl}^{m}(t,\theta,\varphi) = \underbrace{\cos(\frac{2\pi n}{T}t)}_{\text{Fourier series}}\underbrace{Y_l^m(\theta,\varphi)}_{\text{Spherical Harmonics}}$$

Fourier series

Spherical Harmonics

시간적 변화 (Time)

공간적 변화 (Camera view)

Method

- 4D Gaussian

$$\boldsymbol{I(u,v)} = \sum_{i=1}^{N} \boldsymbol{p_i(t)}\boldsymbol{p_i(u,v|t)}\boldsymbol{\alpha_i}\boldsymbol{c_i(d_i,t)} \prod_{j=1}^{i-1} (1 - \boldsymbol{p_i(t)}\boldsymbol{p_i(u,v|t)}\boldsymbol{\alpha_i})$$

$p(t) = N(t; \mu_4, \Sigma_{4,4})$

Time
$p(t)$

4D Gaussian

$$\left\{ \begin{aligned} \mu_{xyzt} &= (\mu_x, \mu_y, \mu_z, \mu_t) \\ R &= L(q_l)R(q_r) \\ s &= (s_x, s_y, s_z, s_t) \end{aligned} \right\}$$

$Z_{nl}^{m}(t,\theta,\varphi) = \cos(\frac{2\pi n}{T} t) Y_l^m(\theta,\varphi)$

Color
$c_i(d_i, t)$

Image
$I(u,v)$

3D Gaussian

2D Gaussian
$p(u,v|t)$

$\Sigma_{xyz|t} = \Sigma_{1:3,1:3} - \Sigma_{1:3,4}\Sigma^{-1}_{4,4}\Sigma_{4,1:3}$

$\mu_{xyz|t} = \mu_{1:3} + \Sigma_{1:3,4}\Sigma^{-1}_{4,4}(t - \mu_t)$

$\Sigma_{2D} = (JE\Sigma E^T J^T)_{1:2,1:2}$

$\mu_{2D} = Proj(\mu|E,K)_{1:2}$

Experiment

- Implementation details
  - Training setup
    - Iteration: 30,000
    - Batch size: 4
  - Initialization
    - Rotation: identity
    - Time scale: half of the scene's duration
- Datasets
  - Real-word dataset
    - Plenoptic Video dataset
      - Multi-view dataset
  - Synthetic dataset
    - D-NeRF dataset
      - Monocular video dataset

Experiment

- Quantitative results

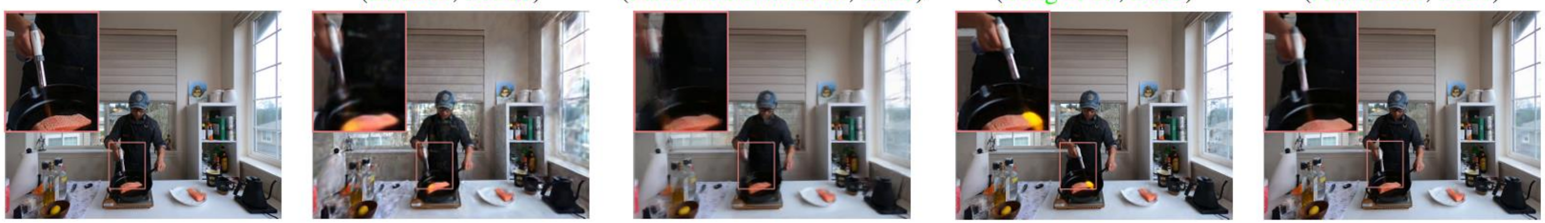
Method	PSNR ↑	DSSIM ↓	LPIPS ↓	FPS ↑
- Plenoptic Video (real, multi-view)				
Neural Volumes (Lombardi et al., 2019)[1]	22.80	0.062	0.295	-
LLFF (Mildenhall et al., 2019)[1]	23.24	0.076	0.235	-
DyNeRF (Li et al., 2022b)[1]	29.58	0.020	0.099	0.015
HexPlane (Cao & Johnson, 2023)	31.70	0.014	0.075	0.56[3]
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**

Experiment

- Qualitative results & Ablation

Ours (114 fps) - | DyNeRF (0.015 fps) (Li et al., 2022b) | K-Planes (0.23 fps) (Fridovich-Keil et al., 2023) | NeRFPlayer (0.045 fps) (Song et al., 2023) | HyperReel (2.00 fps) (Attal et al., 2023)

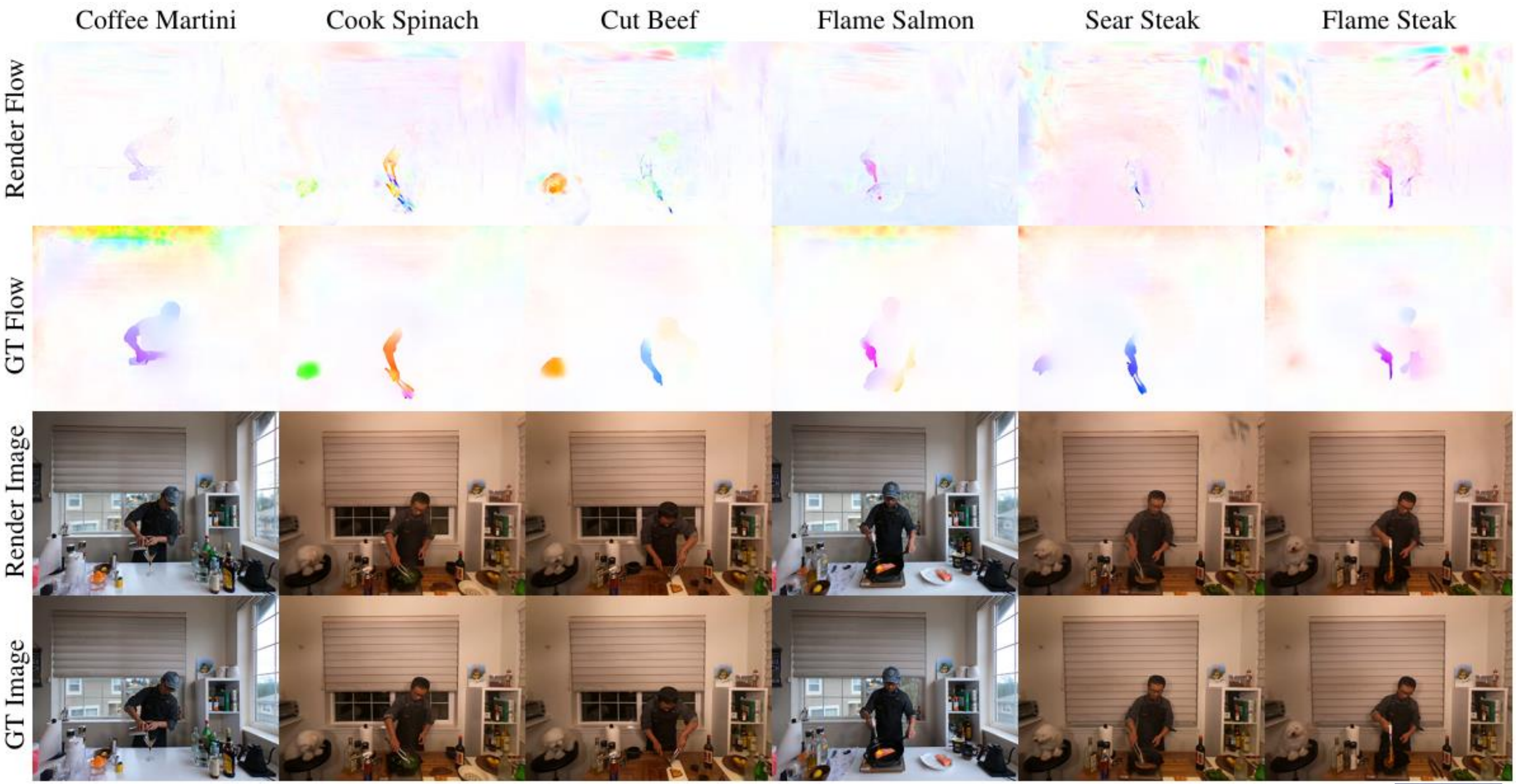


Ground truth - | Neural Volumes (Lombardi et al., 2019) | LLFF (Mildenhall et al., 2019) | HexPlane (0.56 fps) (Cao & Johnson, 2023) | MixVoxels (16.7 fps) (Wang et al., 2023)

	Flame Salmon		Cut Roasted Beef		Average	
	PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑
No-4DRot	28.78	0.95	32.81	0.971	30.79	0.96
No-4DSH	29.05	0.96	33.71	0.97	31.38	0.97
No-Time split	28.89	0.96	32.86	0.97	30.25	0.97
Full	**29.38**	**0.96**	**33.85**	**0.98**	**31.62**	**0.97**

Experiment

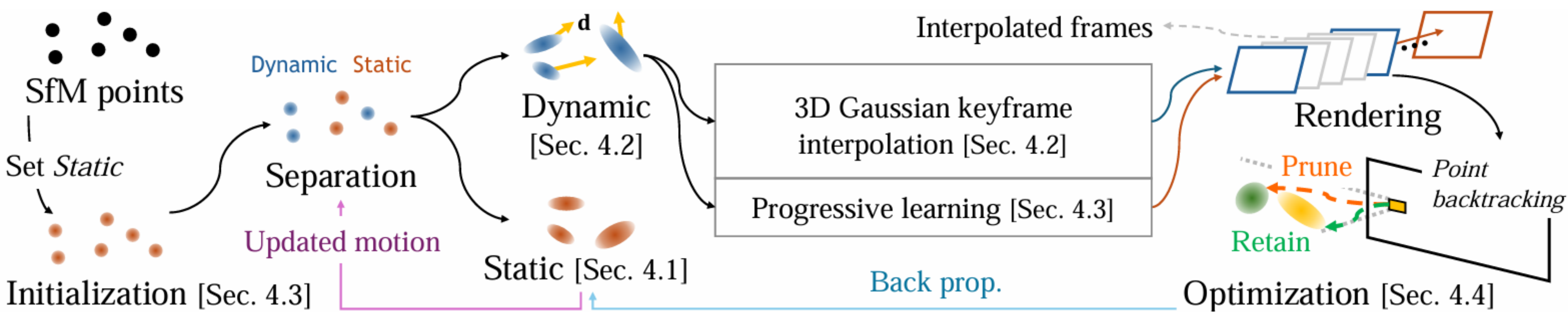
- Public dataset



Fully Explicit Dynamic Gaussian Splatting
NeurIPS 2024

Method

- Overall pipeline



Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Static Gaussian
    - Initialization
      - Scene에서 모든 Gaussian들을 static으로 초기화

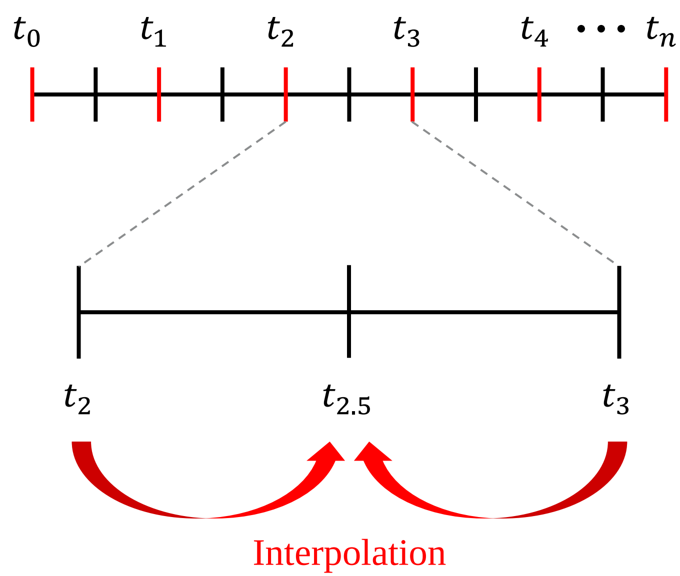
(가정: 모든 static point는 linearly하게 움직임)

$$\boldsymbol{\mu}(\boldsymbol{t}) = \boldsymbol{x} + \underbrace{\boldsymbol{t}'}_{\text{Normalized time}}\underbrace{\boldsymbol{d}}_{\text{Translation}}, \quad \boldsymbol{t}' = \frac{\overbrace{\boldsymbol{t}}^{\text{Time}}}{\underbrace{\boldsymbol{l}}_{\text{Scene 길이}}} \in [\mathbf{0}, \mathbf{1}]$$

Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Dynamic Gaussian
    - Keyframe selection
      - Memory efficient를 위해 모든 frame을 사용하지 않고 keyframe을 선택하여 사용
    - Interpolation
      - Motion의 smooth함을 위해 인접한 두 keyframe들의 각 attribute 별 interpolation을 수행

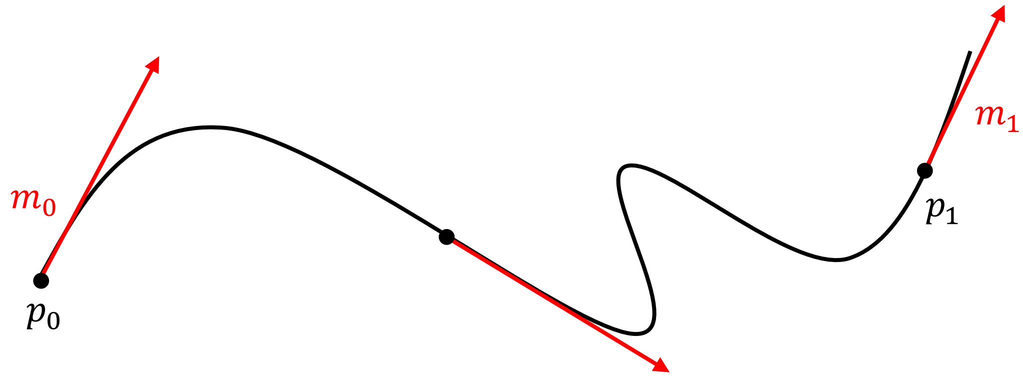
$$K = \{t \mid t = n\overset{\text{Interval}}{I}, n \in \mathbb{Z}, t \in \mathrm{T}\}$$



Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Dynamic Gaussian: Mean
    - Cubic Hermite Interpolator for Temporal Position (CHip)
      - Low-dgree polynomial로 확장하여 overfitting과 over-smoothing을 방지

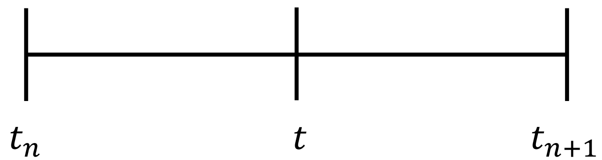
$$\begin{aligned}\mathrm{CHip}(\boldsymbol{p}_0, \boldsymbol{m}_0, \boldsymbol{p}_1, \boldsymbol{m}_1; t) \;=\;& (2t^3 - 3t^2 + 1)\boldsymbol{p}_0 + (t^3 - 2t^2 + t)\boldsymbol{m}_0 \\ & + (-2t^3 + 3t^2)\boldsymbol{p}_1 + (t^3 - t^2)\boldsymbol{m}_1, \qquad \textit{where} \quad t \in [0, 1].\end{aligned}$$



Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Dynamic Gaussian: Mean
    - Cubic Hermite Interpolator for Temporal Position (CHip)
      - Low-dgree polynomial로 확장하여 overfitting과 over-smoothing을 방지

$$\begin{aligned}\mathrm{CHip}(\boldsymbol{p}_0, \boldsymbol{m}_0, \boldsymbol{p}_1, \boldsymbol{m}_1; t) &= (2t^3 - 3t^2 + 1)\boldsymbol{p}_0 + (t^3 - 2t^2 + t)\boldsymbol{m}_0 \\ &+ (-2t^3 + 3t^2)\boldsymbol{p}_1 + (t^3 - t^2)\boldsymbol{m}_1, \qquad \textit{where} \quad t \in [0, 1].\end{aligned}$$



Position of Gaussian at t_n keyframe

Tangent value at t_n keyframe

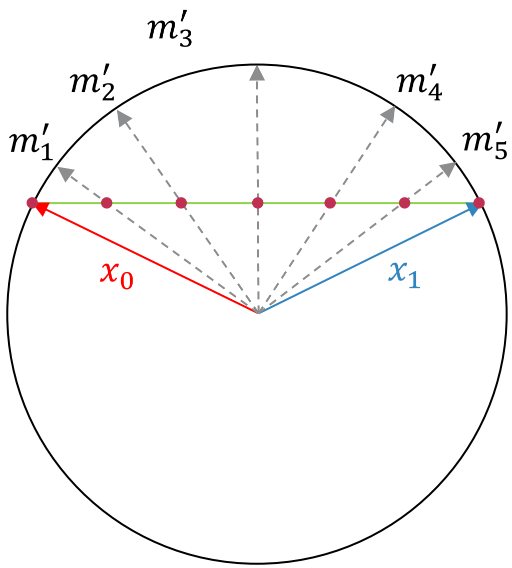
$$\mu(t) = \mathrm{CHip}(\boldsymbol{p}_n, \boldsymbol{m}_n, \boldsymbol{p}_{n+1}, \boldsymbol{m}_{n+1}; t'),$$

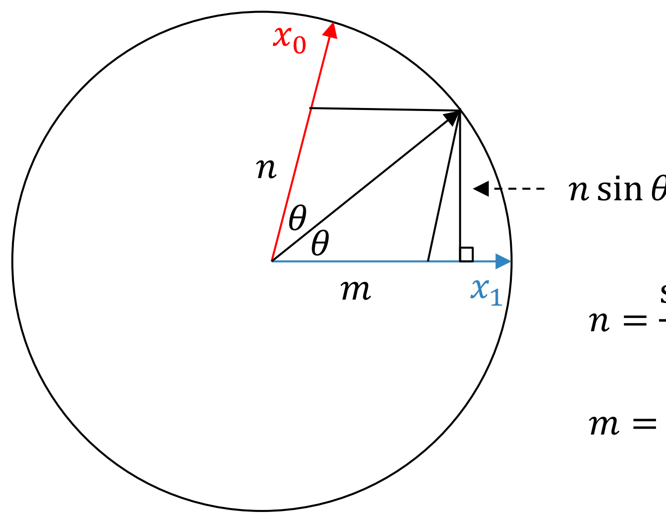
$$\textit{where} \quad n = \left\lfloor \frac{t}{I} \right\rfloor, \quad t' = \frac{t - nI}{I}, \quad \boldsymbol{m}_n = \frac{\boldsymbol{p}_{n+1} - \boldsymbol{p}_{n-1}}{2I}, \quad \boldsymbol{m}_{n+1} = \frac{\boldsymbol{p}_{n+2} - \boldsymbol{p}_n}{2I},$$

Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Dynamic Gaussian: Rotation
    - Spherical Linear Interpolation for Temporal Rotation (Slerp)
      - 회전이나 방향 벡터를 시간에 따라 부드럽게 보간하는 방법

$$\mathrm{Slerp}(\boldsymbol{x}_0, \boldsymbol{x}_1; t) = \frac{\sin[(1-t)\Omega]}{\sin\Omega}\boldsymbol{x}_0 + \frac{\sin[(t)\Omega]}{\sin\Omega}\boldsymbol{x}_1, \quad \textit{where} \quad t \in [0,1] \ \text{ and } \cos\Omega = \boldsymbol{x}_0 \cdot \boldsymbol{x}_1.$$





$$n = \frac{\sin\theta(1-t)}{\sin\theta}$$

$$m = \frac{\sin\theta(t)}{\sin\theta}$$

Method 1. Representation of 4D Gaussian

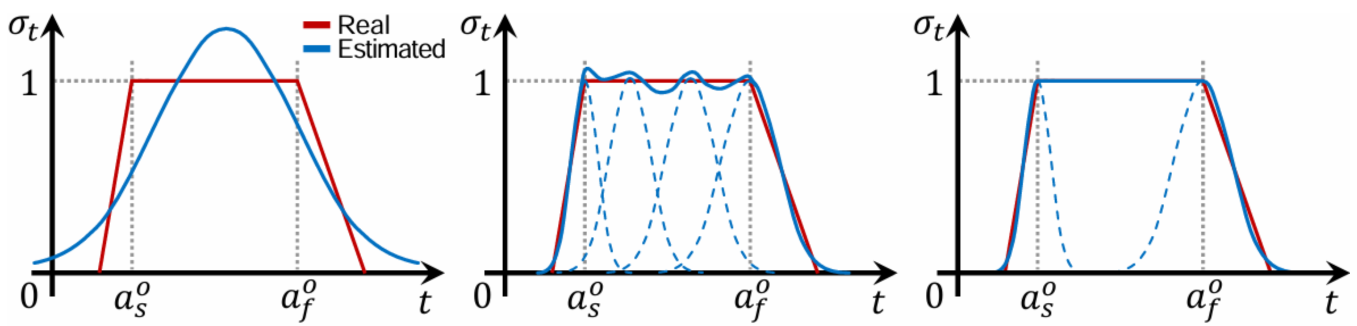
- 4D Gaussian
  - Dynamic Gaussian: Rotation
    - Spherical Linear Interpolation for Temporal Rotation (Slerp)
      - 회전이나 방향 벡터를 시간에 따라 부드럽게 보간하는 방법

$$\mathrm{Slerp}(\boldsymbol{x}_0, \boldsymbol{x}_1; t) = \frac{\sin[(1-t)\Omega]}{\sin\Omega}\boldsymbol{x}_0 + \frac{\sin[(t)\Omega]}{\sin\Omega}\boldsymbol{x}_1, \quad \textit{where} \quad t \in [0,1] \text{ and } \cos\Omega = \boldsymbol{x}_0 \cdot \boldsymbol{x}_1 .$$

$$q(t) = \mathrm{Slerp}(\underbrace{\boldsymbol{r}_n}_{\text{Rotation of Gaussian at } t_n \text{ keyframe}}, \boldsymbol{r}_{n+1}; t') \quad \textit{where} \quad n = \left\lfloor \frac{t}{I} \right\rfloor, \; t' = \frac{t - nI}{I},$$

Method 1. Representation of 4D Gaussian

- 4D Gaussian
  - Dynamic Gaussian: Opacity
    - Simplify Gaussian Mixture for Temporal Rotation
      - 시간 변화에 따라 object의 appearance와 disappearance를 다룸



(a) Single Gaussian (b) Gaussian mixtures (c) **Ours**

a_s^o = Mean of Gaussian
b_s^o = Variance of Gaussian

$$\sigma_t(t) = \begin{cases} e^{-\left(\frac{t-a_s^o}{b_s^o}\right)^2}, & \text{for} \quad t < a_s^o \\ 1, & \text{for} \quad a_s^o \le t \le a_f^o \\ e^{-\left(\frac{t-a_f^o}{b_f^o}\right)^2}, & \text{for} \quad t > a_f^o\,. \end{cases}$$

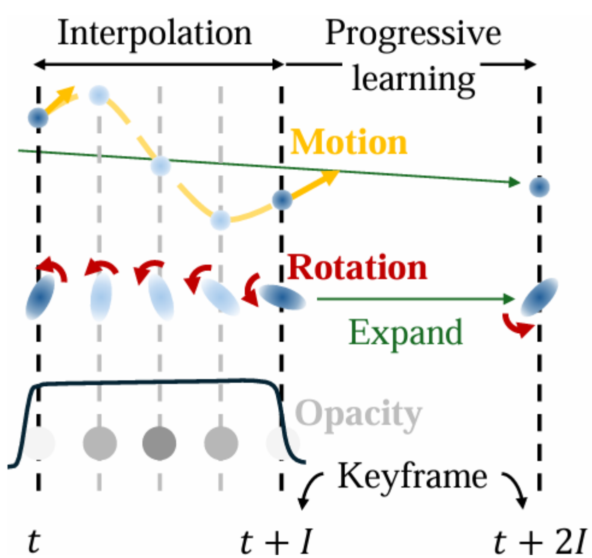
서강대학교
SOGANG UNIVERSITY

VDS LAB

Method 2. Training scheme

- Progressive training scheme
  - First frame으로부터 얻은 소량의 points만을 사용하여 input video의 일부만 학습을 진행
  - 이후 duration을 interval만큼 늘려가며 학습을 진행

- Extracting dynamic points from static points
  - Static point가 움직인 거리를 측정하여 top-n % (In paper, n=2)을 dynamic point로 변환
  - 이때 camera로부터 너무 먼 point들만 선택되는 것을 방지하기 위해 camera와 point 사이의 distance로 normalization을 수행



Experiment

- Implementation details
  - Initialization
    - Time interval: 10
    - Initial duration: 10
    - Increment duration: 400 iteration
  - Dataset
    - Real-world datasets
      - Neural 3D Video dataset
        - ✓Multi-view dataset (grid: 18~21)
      - Technicolor dataset
        - ✓Multi-view dataset (grid: 4x4=16)
  - Evaluation
    - Metric: PSNR, SSIM, LPIPS
    - FPS: including preprocessing

Experiment

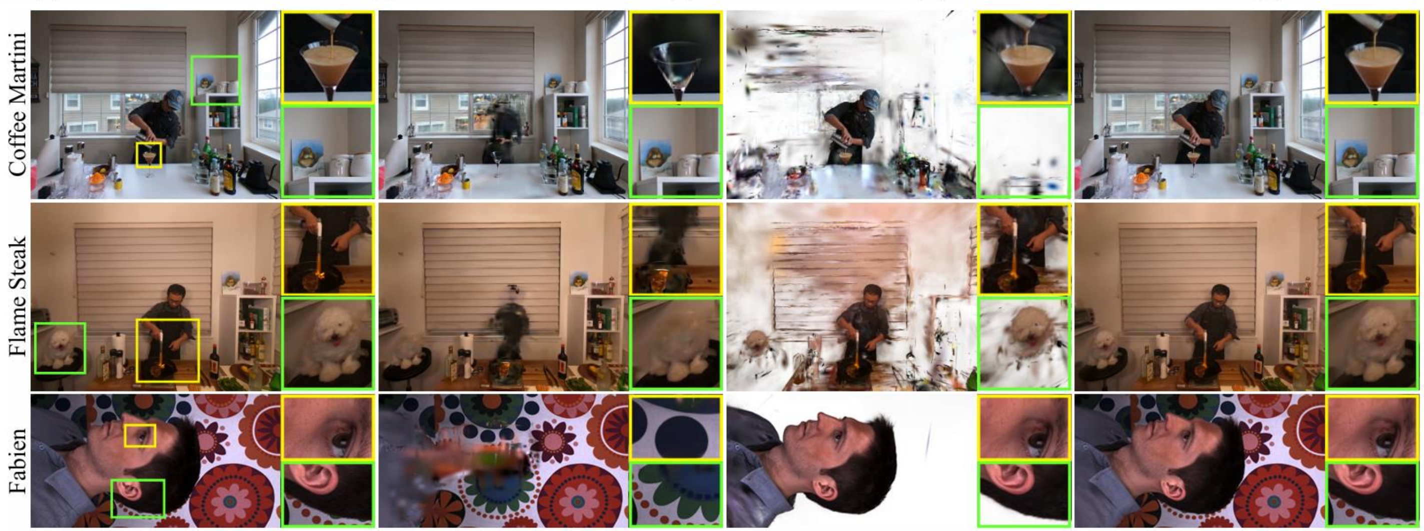
- Neural 3D Video dataset
  - Quantitative results

Model	PSNR (dB)							MB	Frame/s	Hours
	Coffee Martini	Cook Spinach	Cut Roasted Beef	Flame Salmon	Flame Steak	Sear Steak	Average	Size	FPS	Training time
NeRFPlayer [72]	31.53	30.56	29.35	31.65	31.93	29.13	30.69	5130	0.05	6
HyperReel [86]	28.37	32.30	32.92	28.26	32.20	32.57	31.10	360	2	9
Neural Volumes [87]	N/A	N/A	N/A	22.80	N/A	N/A	22.80	N/A	N/A	N/A
LLFF [88]	N/A	N/A	N/A	23.24	N/A	N/A	23.24	N/A	N/A	N/A
DyNeRF [7]	N/A	N/A	N/A	29.58	N/A	N/A	29.58	28	0.015	1344
HexPlane [11]	N/A	32.04	32.55	29.47	32.08	32.39	31.71	200	N/A	12
K-Planes [12]	29.99	32.60	31.82	30.44	32.38	32.52	31.63	311	0.3	1.8
MixVoxels-L [89]	29.63	32.25	32.40	29.81	31.83	32.10	31.34	500	37.7	1.3
MixVoxels-X [89]	30.39	32.31	32.63	30.60	32.10	32.33	31.73	500	4.6	N/A
Im4D [90]	N/A	N/A	32.58	N/A	N/A	N/A	32.58	N/A	N/A	N/A
4K4D [19]	N/A	N/A	32.86	N/A	N/A	N/A	32.86	N/A	110	N/A
Dense COLMAP point cloud input										
STG‡ [15]	28.41	32.62	32.53	28.61	33.30	33.40	31.48	107	88.5	5.2†
4DGS [74]	28.33	32.93	33.85	29.38	34.03	33.51	32.01	6270	71.4	5.5
4DGaussians [14]	27.34	32.46	32.90	29.20	32.51	32.49	31.15	34	136.9	1.7
Sparse COLMAP point cloud input										
STG‡ [15]	27.71	31.83	31.43	28.06	32.17	32.67	30.64	109	101.0	1.3†
4DGS [74]	26.51	32.11	31.74	26.93	31.44	32.42	30.19	6057	72.0	4.2
4DGaussians [14]	26.69	31.89	25.88	27.54	28.07	31.73	28.63	34	146.6	1.5
3DGStream [91]	27.75	33.31	33.21	28.42	34.30	33.01	31.67	1200	-	-
Ours	28.79	33.23	33.73	29.29	33.91	33.69	32.11	115	120.6	0.6

Experiment

- Neural 3D Video dataset
  - Qualitative results

(a) Ground Truth (b) 4DGS (c) STG† (d) 4DGaussians (e) **Ours**



(a) Ground Truth (b) Rendered static points (c) Rendered dynamic points (d) Rendered static&dynamic points

Experiment

- Ablation

Method	PSNR	$SSIM_1$	LPIPS	Size(MB)
w/ Linear position	31.12	0.9385	0.0524	204
w/o Temporal opacity	31.42	0.9394	0.0521	186
w/ Linear rotation	31.26	0.9392	0.0525	148
w/o Progressive growing	31.02	0.9389	0.0550	168
w/ Linear position&rotation	31.32	0.9394	0.0521	172
w/o Regularization	31.37	0.9395	0.0522	174
w/o Dynamic point extraction	28.58	0.9280	0.0756	58
w/o Point backtracking	31.40	0.9394	0.0529	169
Ours	32.11	0.9422	0.0478	115

감사합니다