Recent Advances in Motion Modeling for Dynamic 3D Gaussian Splatting

2026 동계 세미나

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

Presented By
최승규

Outline

- Background
  - Novel View Synthesis
  - 3DGS 이후의 연구
  - 논문 선정 동기

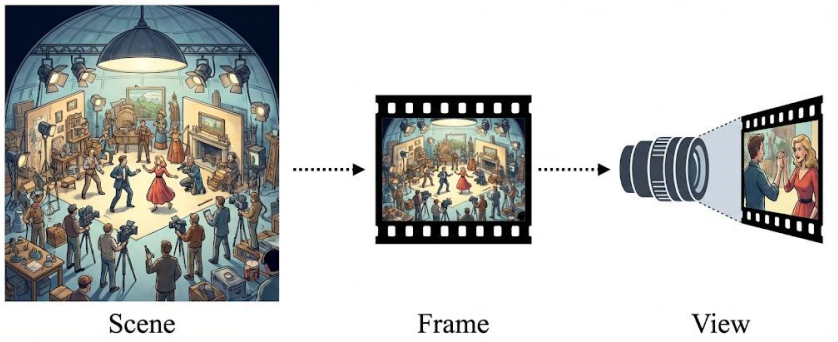
- LocalDyGS: Multi-view Global Dynamic Scene Modeling via Adaptive Local Implicit Feature Decoupling [ICCV 2025]

- ReCon-GS: Continuum-Preserved Gaussian Streaming for Fast and Compact Reconstruction of Dynamic Scenes [NeurIPS 2025]

- Conclusion

Background

- Novel View Synthesis
  - Scene
    - 표현하고자 하는 장면 전체
  - Frame
    - 시간의 흐름에 따른 장면의 순간적인 상태
  - View
    - 특정 위치와 각도에서 장면을 바라보는 카메라의 시점
    - Monocular vs Multi-View
  - Novel View Synthesis:
    - 학습에 사용되지 않은 시점(View)의 장면을 합성해내는 Task



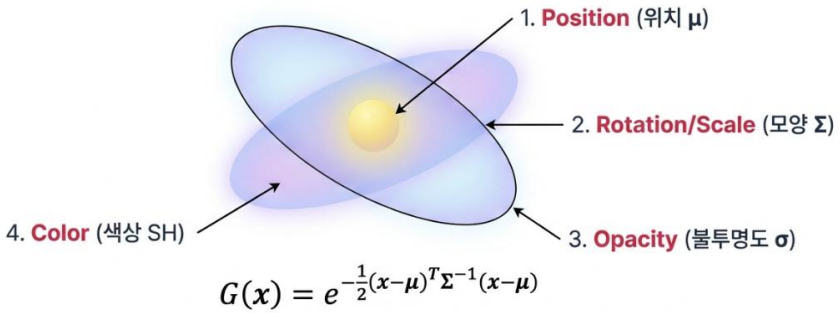
1) Mildenhall et al., NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, ECCV (2020)
2) Kerbl et al., 3D Gaussian Splatting for Real-Time Radiance Field Rendering, ACM Transactions on Graphics (SIGGRAPH) (2023)

Background

- Novel View Synthesis
  - Point Cloud
    - 3D 공간 상의 점들로 장면을 표현
    - 확대하거나 특정 각도에서 보면 점 사이가 비어 형태가 불분명
  - Mesh
    - Point Cloud의 점들을 선으로 연결하여 다각형 면을 형성한 구조
  - NeRF[1)]
    - MLP 신경망 사용
    - 고화질 출력이 가능하지만 학습이나 렌더링 속도가 매우 느림
  - 3D Gaussian Splatting[2)]
    - NeRF의 속도 문제를 해결하며 등장한 게임 체인저 (30 FPS+)
    - 3D 공간에 Gaussian 타원체를 뿌리고 2D로 Projection
    - NeRF와 같은 Ray-Marching 방식 대신 CUDA Rasterizer를 통한 투영 방식 채택
    - 이미지넷 챌린지의 CNN과 같은 혁신

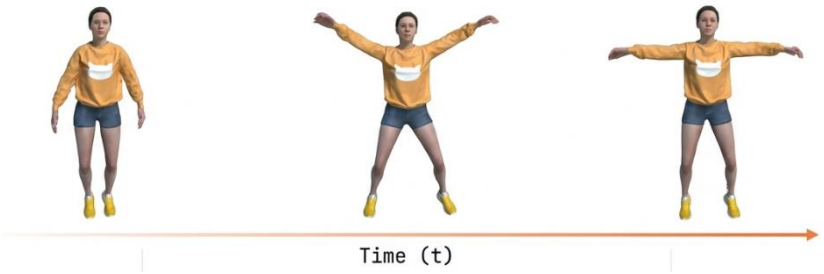
Background

- Novel View Synthesis
  - 3D Gaussian Splatting[1]의 파라미터 구조
    - Position (μ)
      - Gaussian 타원체의 중심 위치
    - Covariance (Σ)
      - Gaussian 타원체의 크기와 회전
      - 대각 행렬 형식의 크기 행렬 S와 회전 quaternion q를 각각 학습 후 결합
    - Opacity (α)
      - 불투명도
    - Color (Spherical Harmonics)
      - 시점 의존적 색상 표현



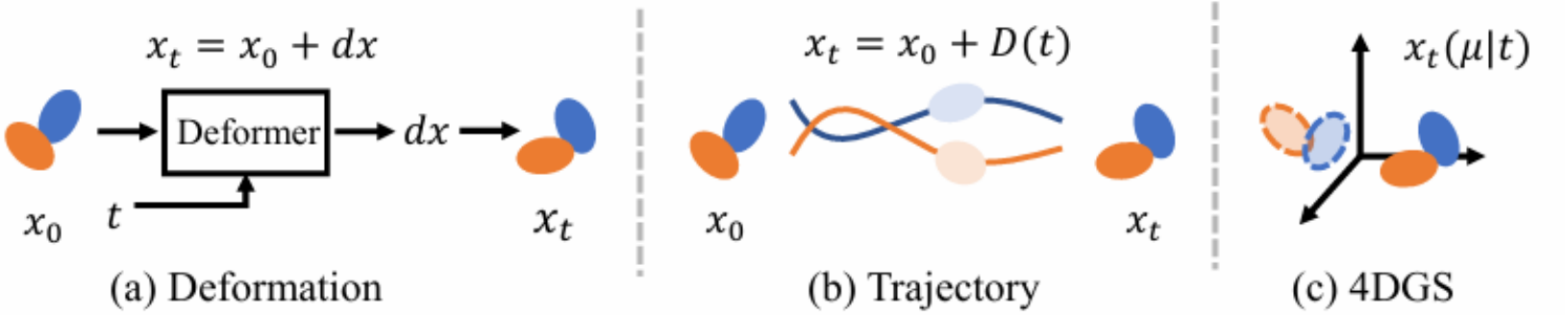
Background

- 3DGS 이후의 연구
  - In-the-wild
    - 다양한 조명 조건 하에서 촬영된 장면의 색감을 통합
  - Continual Learning
    - 이전 학습 내용을 유지하며 새로 추가된 후속 장면의 정보를 통합
  - Compression
    - 렌더링 퀄리티를 유지하며 저장 용량을 줄이고 FPS 향상
  - Dynamic Scene Reconstruction
    - 정적 장면의 재구성에 그치지 않고 시간의 흐름에 따라 변하는 장면을 재구성
    - 오늘 세미나의 주제!



Background

- 논문 선정 동기
  - 단순히 여러 프레임을 따로 학습하면 저장 공간 폭발, 학습 속도 저하
  - Motion에 대한 별도 고려가 없으면 각종 Artifact 발생
  - 최신 논문들이 이 “Motion Representation”을 어떻게 해결했는지 유형 별로 분석



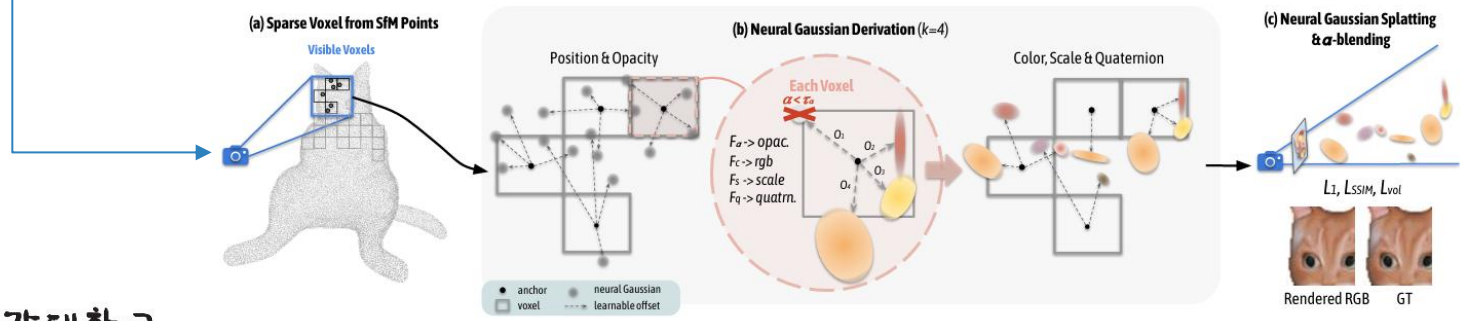
LocalDyGS: Multi-view Global Dynamic Scene Modeling via Adaptive Local Implicit Feature Decoupling [ICCV 2025]

Introduction

- 기존 연구의 문제점
  - Monocular Dynamic Scene Reconstruction
    - 간단한 장면에만 제한적
    - 크게 동적이고 복잡한 장면은 어려움
  - Implicit MLP
    - 단일 MLP를 통한 전체 동적 장면의 표현 시도 -> 용량 부족
    - 크고 복잡한 모션에서 표현력 부족으로 인한 Blurring, Deblurring
  - Explicit Motion Function
    - 각 Gaussian에 다항식 궤적 할당 -> 학습이 매우 느리고 파라미터가 너무 많음
  - 4D Primitives
    - Gaussian 개수가 폭발적으로 증가하여 무거움
- 문제 해결 방법
  - Global Space를 Local Space로 분해
  - 각 Local Space 내부의 움직임 표현을 위해 Temporal Gaussians 생성 (MLP 이용)

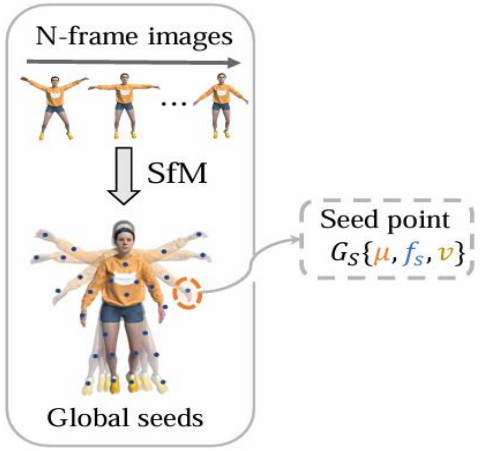
Method

- Preliminary: 3DGS & Scaffold-GS
  - 3DGS
    - 위치, 회전, 크기, 불투명도, 색상을 가지는 3D 가우시안을 2D 화면에 투영
    - $G(x) = e^{-\frac{1}{2}x^T\Sigma^{-1}x}$
  - Scaffold-GS[1)]
    - Anchor를 기준으로 Neural Gaussian을 생성
    - 정적 장면에 한정
  - LocalDyGS의 차별점
    - 정적 장면의 재구성에 그치지 않고 Anchor 개념을 시공간으로 확장
    - Seed를 통해 동적 장면을 효율적으로 표현



Method

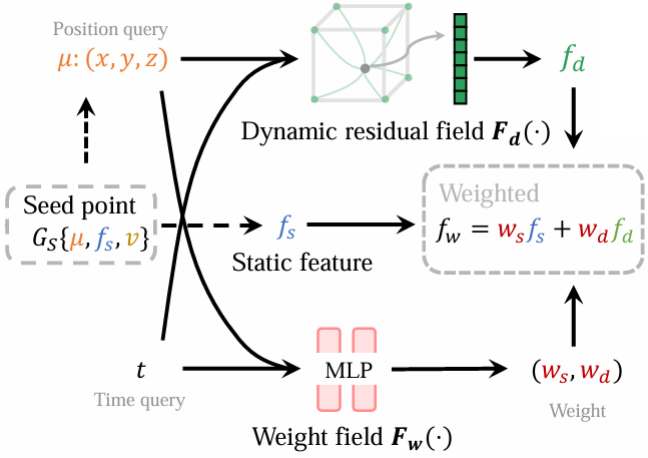
- Global Seeds Initialization
  - Local Space를 커버하는 Seed의 위치 초기화
    - 초기 프레임의 SfM 포인트 클라우드를 병합, 초기 Seed 위치로 사용
    - 물체가 나타날 수 있는 곳에 대한 사전 정보 제공 역할
  - Seed 파라미터
    - Seed는 시간이 흘러도 변하지 않는 정적 정보를 포착, $F_s \in R^{64}$ 저장
    - Local Space의 커버 범위를 나타내는 스케일 파라미터 $v \in R^3$
    - Seed 별로 위치 μ, 정적 특징 F_s , 스케일 파라미터 v 저장
  - 스케일 파라미터 v의 초기화
    - 가장 가까운 3개 Seed까지의 거리를 평균
    - Seed 간 거리에 비례해 커버 범위 설정



Methods

- Feature-Decoupled Spatio-Temporal Fields
  - 정적 정보와 동적 정보를 분리하여 효율적으로 모델링
  - 단일 MLP로 시공간을 한 번에 배우려고 하면 용량 한계로 artifact 발생
    - 정적인 특징과 동적인 특징을 분리하여 학습 후 병합
  - Multi-Resolution 4D Hash Encoding
    - 입력: 위치 μ, 시간 t
    - 출력: $h_{4d}(\mu, t; l) \in R^m \; for \; L \; scales$
    - $f_h(\mu, t) = [h_{4d}(\mu, t; 1), \ldots, h_{4d}(\mu, t; L)]$
  - Dynamic Residual Field (F_d)
    - 입력: $f_h(\mu, t)$
    - 출력: 동적 잔차 특징 f_d
  - Adaptive Weight Field (F_W)
    - 입력: f_s, f_d
    - 출력: 두 Feature의 가중치 w_s, w_d
  - 최종 특징: $F_W = w_s * f_s + w_d * f_d$

(a) Static feature (b) Dynamic feature (c) Weighted feature



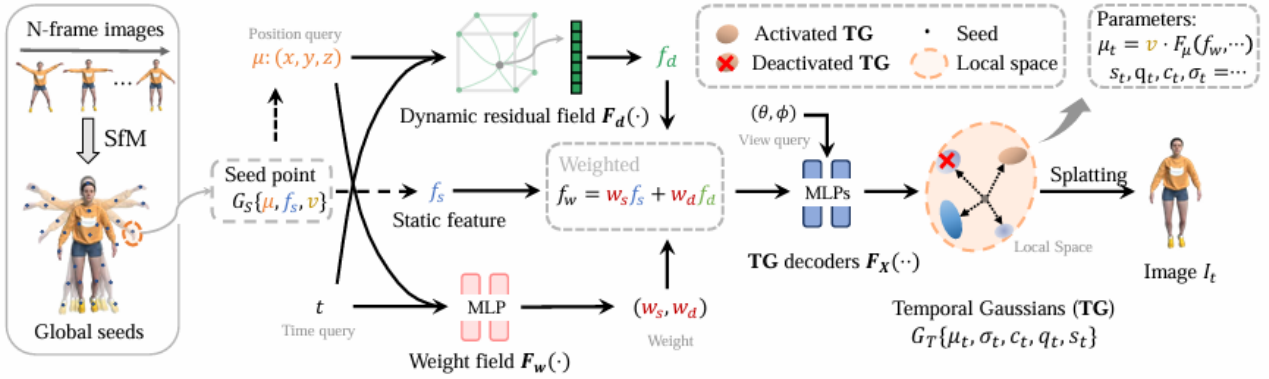
Methods

- Local Temporal Gaussian Derivation
  - 결합된 특징을 바탕으로 실제 렌더링 가능한 Temporal Gaussians 생성
  - 2-layer MLP가 Gaussian 특성 별로 존재
    - Hyperparameter K를 통해 Seed 당 출력 Gaussian 수 조정
    - Gaussian 별로 μ, q, s, α, c의 전체 파라미터를 출력

$$\{u_t^i\}_{i=0}^{k-1} = \mu + v * F_u(f_w)$$

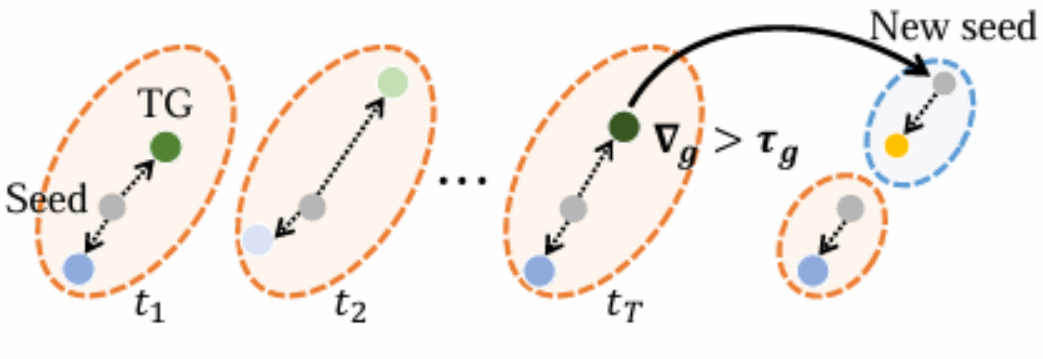
$$\{\sigma_t^i\}_{i=0}^{k-1} = Sigmoid\big(F_0(f_w, d)\big),\ d = \frac{\mu - \mu_c}{\|\mu - \mu_c\|_2}$$

  - Deactivation Strategy
    - 출력된 Gaussian 중 불투명도가 Threshold 이하인 Gaussian은 렌더링 시 제외



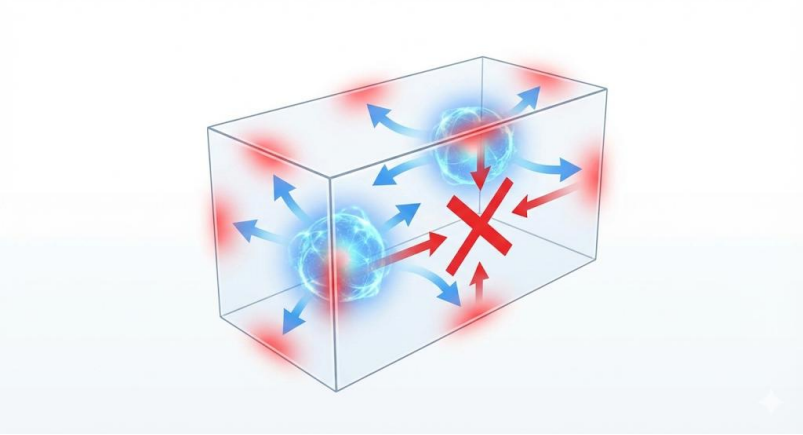
Methods

- Adaptive Seed Growing (ASG)
  - 초기 포인트 클라우드의 불완전함을 학습 중 보완
  - 초기 SfM 포인트가 없는 곳은 Seed가 존재하지 않고 텅 비어 있음
    - 기울기가 Threshold보다 큰 Gaussian은 위치가 불안정하다고 판단
    - 해당 위치에 새로운 Seed를 추가하고 Local Space를 생성
    - 빈 공간이 점진적으로 채워지며 렌더링 품질 향상



Methods

- Loss Function
  - 3DGS의 Loss 그대로 차용
    - L_1 Loss: 픽셀 간 단순 차이를 고려
    - L_{SSIM} Loss : 시각적 유사도를 고려
  - Volume Regularization
    - Gaussian이 불필요하게 커져 다른 Local Space를 침범하는 것을 방지
    - $L_v = \sum_{i=1}^{M} Prod(s_t^i)$
  - 최종 Loss
    - $L = (1 - \lambda) * L_1 + \lambda * L_{SSIM} + \lambda_v * L_v$



Experiments

- Implementation
  - RTX3090, 30,000 iterations
  - K = 10, MLP 2-layer + ReLU
  - f_s 64 dimensions, hash size 2^{17}
  - ASG는 3,000 iter부터 15,000 iter까지, 100iter마다 수행
  - $\tau_g = 0.001,\ \tau_\alpha = 0.01,\ \lambda_{SSIM} = 0.2,\ \lambda_{vol} = 0.01$
- Datasets
  - N3DV
    - 2704x2028, 30FPS
  - MeetRoom
    - 1280x720, 30FPS
  - VRU Basketball Court
    - 1920x1080, 25FPS

Experiments

- Quantitative Comparisons
  - N3DV
    - Offline/Online 모두 비교
    - 기존 SOTA 대비 속도 10배, 저장 공간 절반
  - MeetRoom
    - 일반화 성능 검증
    - 저장 공간 적고 퀄리티 우수
  - VRU Basketball Court
    - Large Scale Motion 표현도 훌륭

Table 1. **Quantitative comparisons on the Neural 3D Video Dataset [25].** "Size" is the total model size for 300 frames. $DSSIM_1$ sets data range to 1.0 while $DSSIM_2$ to 2.0 [26]. $*$ indicates online method.

Method	PSNR↑	$DSSIM_1$↓	$DSSIM_2$↓	LPIPS↓	FPS↑	Time↓	Size↓
StreamRF * [23]	28.26	-	-	-	10.9	-	5310 MB
NeRFPlayer [38]	30.69	0.034	-	0.111	0.05	6.0 h	5130 MB
HyperReel [1]	31.10	0.036	-	0.096	2	-	360 MB
K-Planes [14]	31.63	-	0.018	-	0.3	5.0 h	311 MB
HexPlane [9]	31.70	-	0.014	0.075	0.21	12.0 h	240 MB
MixVoxels [42]	31.73	-	0.015	0.064	4.6	-	500 MB
4DGaussian [47]	31.02	0.030	-	0.150	30	0.67 h	90 MB
3DGStream[1] [39]	31.67	-	-	-	215	1.0 h	1230 MB
RealTimeGS [55]	32.01	-	0.014	0.055	114	9.0 h	> 1000 MB
SpaceTimeGS [26]	32.05	0.026	0.014	0.044	140	> 5 h	200 MB
LocalDyGS(Ours)	32.28	0.028	0.014	0.043	105	0.58 h	100 MB

Table 2. **Quantitative comparison on the MeetRoom dataset [23].** PSNR is averaged across all 300 frames, while training time and storage requirements accumulate over the entire sequence.

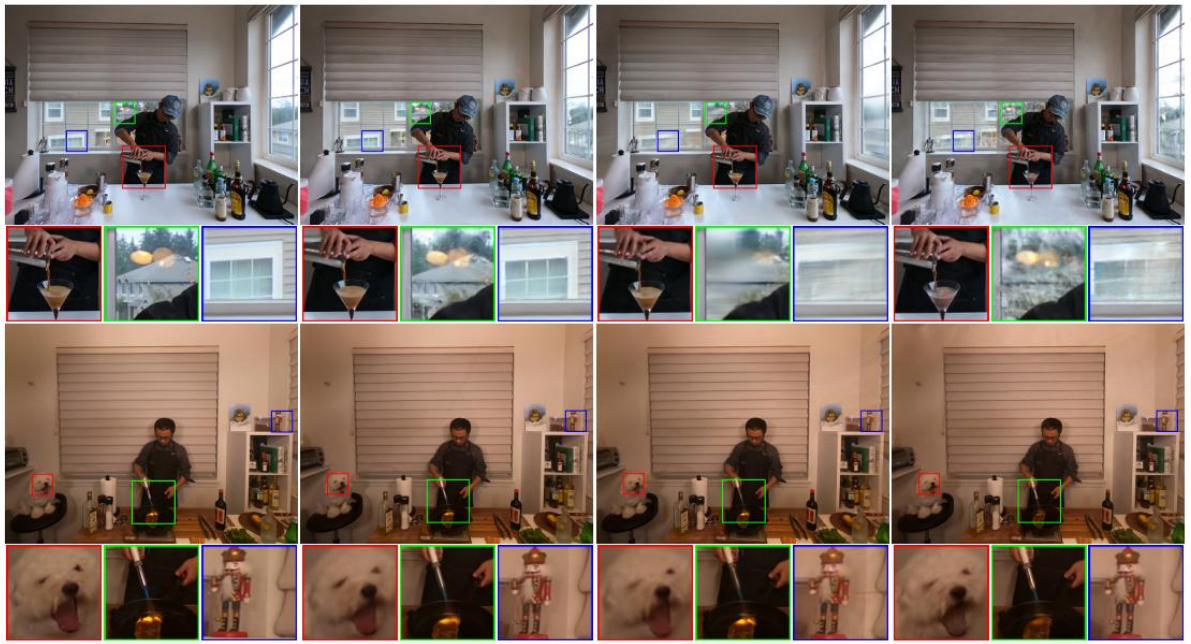
Method	PSNR ↑	Time(hours) ↓	Size(MB) ↓
Plenoxel [13]	27.15	70	304500
I-NGP [31]	28.10	5.5	14460
3DGS [20]	31.31	13	6330
StreamRF [23]	26.72	0.85	2700
3DGStream [39]	30.79	0.6	1230
LocalDyGS(Ours)	**32.45**	**0.36**	**90**

Table 3. **Quantitative comparison on VRU (GZ) basketball court dataset [40].** Static methods are tested on frame 0.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
GOF [58]	30.39	0.949	0.141
2DGS [18]	30.78	0.949	0.187
3DGS [20]	30.50	0.949	0.171
4DGS [47]	28.32	0.930	0.186
SpaceTimeGS [26]	27.42	0.926	0.193
LocalDyGS(Ours)	**30.58**	**0.944**	**0.173**

Experiments

- Qualitative Comparisons
  - Streaming / Non-Streaming 가리지 않고 Large Scale Motion 성능 우위
  - 손처럼 복잡한 부분도 잘 모델링



(a) GT (b) Ours (c) SpaceTimeGS [26] (d) 3DGStream [39]

Experiments

- Ablation Study
  - Static Feature 없으면 성능/학습효율 크게 하락 -> 정적/동적 특징 분리가 큰 역할
  - ASG 또한 재구성 정확도 개선에 큰 역할
  - Seed 초기화를 위한 프레임의 수 N은 6에서 최상
  - MLP의 출력 Gaussian 수 K는 10이 최상

Table 5. **Ablation study of proposed components.** Conducted on the N3DV dataset [24].

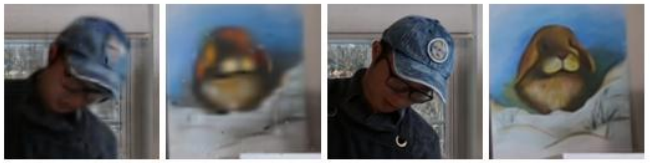
Method	*Coffee*	*Sear Steak*	Mean ↑	FPS ↑	Time ↓
w/o static	26.24	32.68	29.46	96	42 mins
w/o deactivation	29.20	33.18	31.19	89	40 mins
Full	29.03	33.77	31.40	105	36 mins

Table 6. Ablation study on the number of frames whose SfM point clouds are used in initialization, conducted on N3DV.

N Frames	PSNR↑	$DSSIM_1$↓	LPIPS↓	Time↓
N = 30	32.30	0.028	0.043	0.75 h
N = 6	32.28	0.028	0.043	0.58 h
N = 1	31.84	0.041	0.058	0.52 h

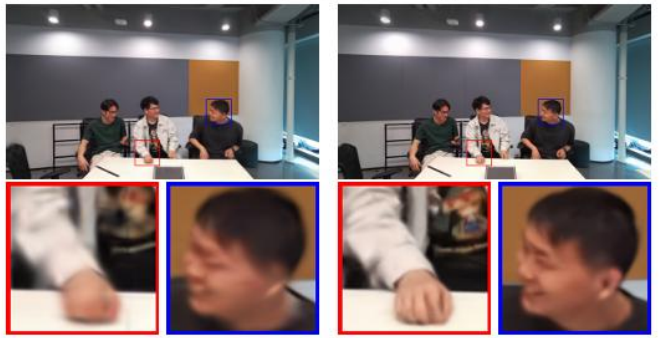
Table 7. Ablation study on different values of k (N3DV dataset).

k value	PSNR↑	DSSIM↓	Time↓	FPS↑
k = 5	32.15	0.028	31.2 m	118
k = 10	32.28	0.028	34.8 m	105
k = 20	31.96	0.032	40.5 m	86



(a) w/o static feature in training (b) Full

Figure 9. A comparison of (a) to (b) shows that training using only dynamic features leads to significant blurring issues.



(a) w/o ASG (b) w/ ASG

Figure 10. Ablation study conducted on the *discussion* scene.

ReCon-GS:
Continuum-Preserved Gaussian Streaming for Fast and Compact Reconstruction of Dynamic Scenes [NeurIPS 2025]

Introduction

- 기존 연구의 문제점
  - Online / Streaming
    - 실시간 처리는 가능하지만 오차가 누적됨 (Drift)
  - Deformation Field
    - 내재되는 체계적 물리적 특성을 무시, uniform encoding -> 부정확한 모델링
    - 반복적 저장으로 파라미터 낭비, 오류 누적으로 정확도 저하
  - 보조적 메커니즘의 활용 (Optical Flow 등)
    - 학습 비용이 크고, 실제 적용을 위해서는 성능 타협이 필요
    - 저장 효율과 성능을 동적으로 조정 불가능, 정적인 프레임워크
- 해결 방법
  - N_{anchor} 조절을 통한 동적인 저장 효율-성능 관리
  - Motion을 Coarse-to-fine으로 3단계 분해
  - 주기적으로 Anchor를 재설정하여 Drift 방지
  - Anchor가 바뀌어도 기존의 기하 정보를 수학적으로 상속

Method

- Overview
  - Base Gaussians
    - 첫 번째 프레임을 통해 정적 3DGS 장면 학습
    - 위치 정보에 노이즈를 섞어 Gaussian들이 더 고르게 분포하게 유도
  - Adaptive Hierarchical Motion Representation
    - 각 Gaussian의 움직임은 여러 기하학적 스케일의 layer-wise 강체 변환으로 파라미터화
    - 각 layer는 grid-based farthest-point sampling으로 분포된 Anchor Gaussian에 의해 통제
    - General Gaussian은 가장 가까운 Anchor에게서 모션 파라미터를 상속
  - Dynamic Hierarchy Reconfiguration
    - 변화하는 장면 기하에 상응하도록 Anchor Gaussians 재배치
    - 계층 내 Deformation 상속을 통해 시간에 따른 움직임의 일관성은 유지
    - View-Adaptive Densification을 통해 렌더링 충실도 향상

Method

- Adaptively Hierarchical Motion Representation
  - Anchor Gaussian Initialization

$$M = \left\lceil N_{anchor}^{1/3} \right\rceil$$

$$\mathbf{g}_a = \arg\min_{\mathbf{g} \in \mathcal{C}_{i,j,k}} \|\boldsymbol{\mu} - \mathbf{c}_{i,j,k}\|_2, \quad where \ \mathbf{c}_{i,j,k} = \boldsymbol{\mu}_{\min} + \left(i + \frac{1}{2}, j + \frac{1}{2}, k + \frac{1}{2}\right) \odot \frac{\boldsymbol{\Delta}}{M}$$

  - 공간을 그리드로 나누고, 각 셀 중심에 가장 가까운 Gaussian을 Anchor로 선정
  - Hierarchical Rigid-Cluster Formation

$$M = \left\lceil (N_{anchor} \cdot 3^{l-1})^{1/3} \right\rceil$$

$$g_a^* = \arg\min_{g_a \in \mathcal{G}_A} \|\mu_n - \mu_a\|_1$$

  - 하위 레벨에서 상위 레벨로 갈 수록 촘촘하게 그리드를 나눔 (Coarse-to-fine)
  - 각 Gaussian은 Anchor이거나, 가장 가까운 Anchor에 종속되는 General Gaussian
  - Explicit Motion Composition

$$\Delta\boldsymbol{\mu}_h = \sum_{l=1}^{3} \Delta\boldsymbol{\mu}^{(l)}, \quad \Delta\boldsymbol{q}_h = \sum_{l=1}^{3} \Delta\boldsymbol{q}^{(l)}$$

  - MLP 통과 없이 단순 덧셈 연산, 빠른 속도
  - “미소 변화량”으로서 정의하고 학습하므로 quaternion 덧셈에 문제가 없음

Method

- Dynamic Hierarchy Reconfiguration
  - Periodic Re-Hierarchization
    - Streaming 방식 특유의 Drift 문제를 해결하기 위함
    - 주기적으로 현 상태의 Gaussian 위치를 반영하여 Anchor 재선정
  - Intra-hierarchical Deformation Inheritance
    - 새로운 Anchor는 초기화 시 변형 파라미터 값이 0이므로 이전 기하 정보 상속 필요
    - 이동 상속
      - 단순 좌표 값이므로 단순 합산 $\Delta\boldsymbol{\mu}_a'^{(l)} = \frac{1}{3}\sum_{i=1}^{3}\Delta\boldsymbol{\mu}_{a_i}^{(l)}$
    - 회전 상속
      - quaternion은 Double Cover 문제로 단순 평균 불가
      - 세 quaternion의 "가장 지배적인 방향" 계산 -> M의 가장 큰 λ 가지는 고유벡터
      - 계산 증명을 위해 레일리 몫, 라그랑주 상수 개념 필요

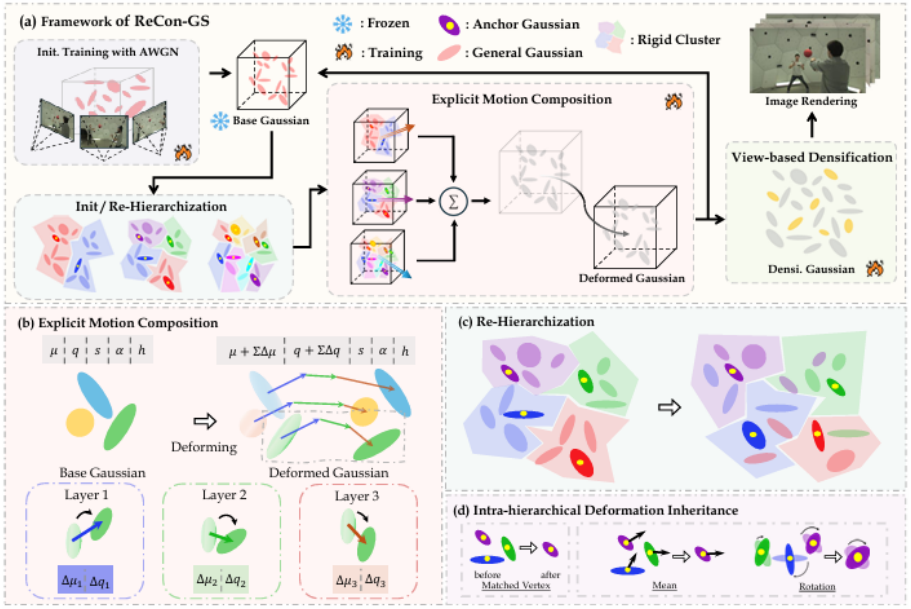
$$\Delta\boldsymbol{q}_a'^{(l)} = \frac{v_{\max}(\mathcal{M})}{\|v_{\max}(\mathcal{M})\|}, \quad \mathcal{M} = \sum_{i=1}^{3}\Delta\boldsymbol{q}_{a_i}^{(l)}\left(\Delta\boldsymbol{q}_{a_i}^{(l)}\right)^{\mathsf{T}}$$

Method

- Intra-hierarchical Deformation Inheritance
  - 회전 상속
    - Quaternion의 Double Cover 현상을 고려하여 M 설정 $\mathcal{M} = \sum_{i=1}^{3} \Delta \boldsymbol{q}_{a_i}^{(l)} (\Delta \boldsymbol{q}_{a_i}^{(l)})^{\mathsf{T}}$
    - M의 방향을 가장 잘 나타내는 quaternion을 상속하는 것이 목적
      - $v^T Mv$를 최대화하는 v $(subject\ to \parallel v \parallel^2 = v^T v = 1)$를 찾아야 함
    - 이 때 최대화하고자 하는 표현 $v^T Mv$에 대하여, $v^T v = 1$이므로 $\mathrm{R}(M, v) = \frac{v^T Mv}{v^T v} = v^T Mv$
      - 레일리 몫을 최대화하는 벡터 v를 구하는 것과 같은 상황이 됨
    - 목적 함수 $\mathrm{f(v)} = v^T Mv$, 제약 조건 $g(\mathrm{v}) = v^T v - 1 = 0$으로 두면,
      - 라그랑주 함수 $\mathrm{L}(v, \lambda) = v^T Mv - \lambda(v^T v - 1)$
      - $\frac{\partial L}{\partial v} = \frac{\partial}{\partial v}(v^T Mv) - \lambda \frac{\partial}{\partial v}(v^T v) = 2Mv - 2\lambda v$
      - $2Mv - 2\lambda v = 0 \rightarrow Mv = \lambda v$ (v는 M의 eigenvector)
    - 다시 처음의 $v^T Mv = v^T \lambda v$인데 $v^T v = 1$이므로, 최대화하려는 값은 λ값에 비례함
    - 따라서, v는 λ를 최대화하는 M의 eigenvector

Method

- Storage-aware Optimization
  - Density Control
    - 사용 가능한 앵커의 수에 따라 밀도 조절 가능
  - Two-Phase Optimization
    - Phase 1: Deformation Field만 학습, Base Gaussian의 기하 및 색상은 고정
    - Phase 2: View-Based Densification을 통해 학습되지 않은 움직임을 다룸



Experiments

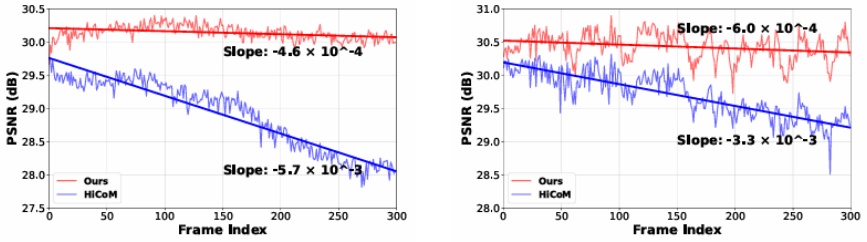
- Dataset
  - N3DV
    - 2704x2028, 30FPS
  - Meeting Room
    - 1280x720, 30FPS
  - PanopticSports
    - 640x360, 30FPS
  - Technicolor
    - 2048x1088, 30FPS
- Implementation Details
  - RTX 4090, 10,000iter ~ 15,000iter
  - 첫 프레임 재구성은 SH를 1로 설정, $\lambda_{noise} = 0.01$,
  - 가장 fine한 level의 Gaussian 수는 전체의 $\frac{1}{24}$, coarse 단계로 갈 수록 $\frac{1}{3}$배
  - 초기 100 iter는 Deformation Field 학습

Experiments

- Quantitative Comparisons
  - 화질 및 렌더링 품질
    - Offline 모델 포함에도 SOTA 달성
    - 기존 Online 방식 대비 PSNR 0.5dB 이상 개선
  - 저장 효율성
    - PanopticSports 기준, 이전 Streaming 방식 대비 저장 용량 60%
    - 비교된 Offline 모델들은 Implicit Motion 표현으로 저장 용량이 유리
      - ReCon-GS가 조금 더 많은 메모리 사용
  - 시간적 안정성
    - Drift에 의한 성능 저하 문제 효과적 해결

Table 3: **Quantitative comparison** on Technicolor dataset. The storage metric includes the size with the initial frame. The training time metric includes the first frame training.

Category	Method	PSNR (dB)↑	SSIM↑	Storage (MB)↓	Render (FPS)↑
NeRF-based	HyperReel [49]	31.80	0.906	1.20	4
Offline	STG [9]	33.60	-	1.10	87
	Ex4DGS [34]	33.62	0.916	2.81	72
Online	E-D3DGS [50]	33.24	0.907	1.54	79
	ReCon-GS (ours)	33.83	0.932	0.82	207



(a) *Coffee Martini* (b) *Flame Salmon*

Figure 3: The PSNR Trend Comparison between Ours and HiCoM [3] across different scenarios.

Experiments

- Qualitative Comparisons
  - N3DV 등에서 기존 Streaming 모델[1, 2] 대비 Motion 표현 우수
  - Hierarchical Level은 3에서 가장 우수

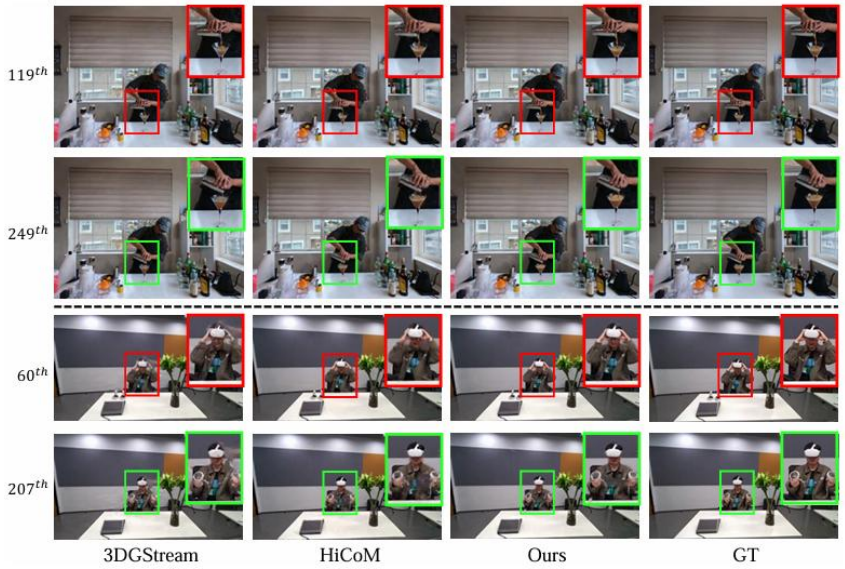


Figure 4: **Qualitative results** on the *coffee martini* scene in the N3DV Dataset. To ensure a fair comparison, we retrained their official code with the same initial sparse points.

Figure 5: Qualitative results of our ReCon-GS under different hierarchical levels

Experiments

- Ablation Study
  - Hierarchical Motion Representation은 메모리 Overhead는 있으나 성능 향상이 큼
  - Dynamic Hierarchy Reconfiguration, View-based Densification 또한 성능에 기여
  - 계층 깊이는 3 Level에서 가장 효율적

Table 4: **Ablation on key components of our ReCon-GS framework.**

Method	PSNR (dB)↑	SSIM ↑	Storage (MB)↓	Train (sec)↓	Render (FPS) ↑
w/o Hierarchical Motion Representation	31.43	0.9525	**0.32**	**6.37**	260
w/o Dynamic Hierarchy Reconfiguration	32.00	0.9521	0.45	6.45	248
w/o View-based Densification	32.15	0.9559	0.40	6.52	**287**
Ours (full)	**32.66**	**0.9571**	0.44	6.44	250

Table 5: **Ablation on hierarchy depth of our ReCon-GS framework.**

Hierarchy Depth	N3DV			Meet Room		
	PSNR (dB) ↑	Storage (MB) ↓	Train (sec) ↓	PSNR (dB) ↑	Storage (MB) ↓	Train (sec) ↓
2	32.422	**0.40**	**6.44**	30.20	**0.28**	3.87
3	**32.662**	0.44	6.49	30.84	0.30	**3.86**
4	32.658	0.46	6.54	**31.01**	0.32	4.01

Conclusion

- 결론
  - 두 논문 모두 기존 방법론의 기조를 유지하면서 한계를 극복함
    - LocalDyGS의 경우, 단일 MLP가 전체 motion을 커버하지 못함을 인지
      - Seed 구조와 Static/Dynamic 분리 등을 통해 각 MLP의 load를 줄여 해결
    - ReCon-GS의 경우, explicit한 표현이 메모리 문제를 야기함을 지적
    - 또한 Streaming 조건에서의 Drift 발생을 예방하고자 함
      - Hierarchical Anchor 표현을 통해 저장 요소를 절감, tradeoff 조절 가능
      - Periodic Hierarchization을 통해 Drift 완화

- 한계
  - Threshold 설정에 대한 수치적 추정 과정이 부족, Monocular 대응 불가
  - LocalDyGS의 경우 각 MLP의 구조에 대한 기술이 없음, SfM 의존성 문제
  - ReCon-GS의 경우 Motion Composition 수식 증명이 없고, 출현/소멸 유연성 부족

감사합니다