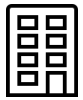
CVSP 겨울 세미나

3D Representation for Dynamic scenes



Pusan National University
Dept. of Electronics Engineering
Computer Vision & Signal Processing Lab.



Presented By
석사과정 김태훈

Table of Contents

- Dynamic Scene Reconstruction
- FreeTimeGS: Free Gaussian Primitives at Anytime and Anywhere for Dynamic Scene Reconstruction [CVPR 2025]
  - Background
- 4DGT: Learning a 4D Gaussian Transformer Using Real-World Monocular Videos [NeurIPS 2025 Spotlight]

Tasks

* Ho, Jonathan, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models." Advances in neural information processing systems 33 (2020): 6840-6851.

- Dynamic Scene Reconstruction (4D Reconstruction)
  - Spatial Reconstruction (3D) + Temporal Reconstruction (1D)



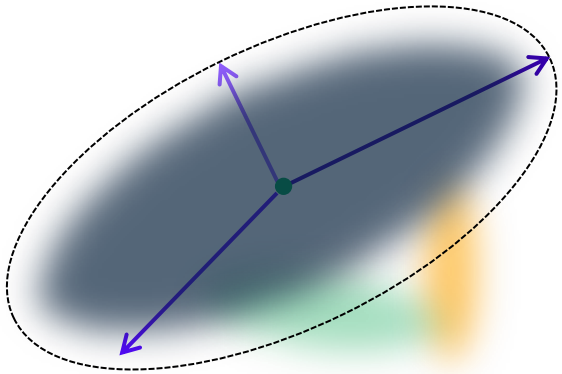
FreeTimeGS:
Free Gaussian Primitives at Anytime and Anywhere for Dynamic Scene Reconstruction
[CVPR 2025]

Background

- Gaussian function
  - 두 가지 파라미터 μ, σ를 이용해 모든 함수 개형을 표현할 수 있다.
  - μ : 중심 위치를 조정함
  - σ : 중심에서의 반경을 조정함

Background

- Static Gaussian Primitive (3DGS)[1]
  - 3D Gaussian을 이용한 삼차원 표현 방법
    - 삼차원 Gaussian function 또한 두 개의 파라미터 μ, Σ로 모든 함수 개형을 나타낼 수 있음
    - 추가적으로 더 좋은 표현력을 위해 투명도 α와 색상 c를 추가하여 표현함



$\mu = (0.2312, 0.6583, 0.1480)$

$\Sigma = (축_1, 축_2, 축_3)$

$\alpha = 0.85$

$c = Gray$

$$G(\boldsymbol{x}) = \alpha \cdot \exp\left(-\frac{1}{2}(\boldsymbol{x}-\mu)\Sigma^{-1}(\boldsymbol{x}-\mu)\right)$$

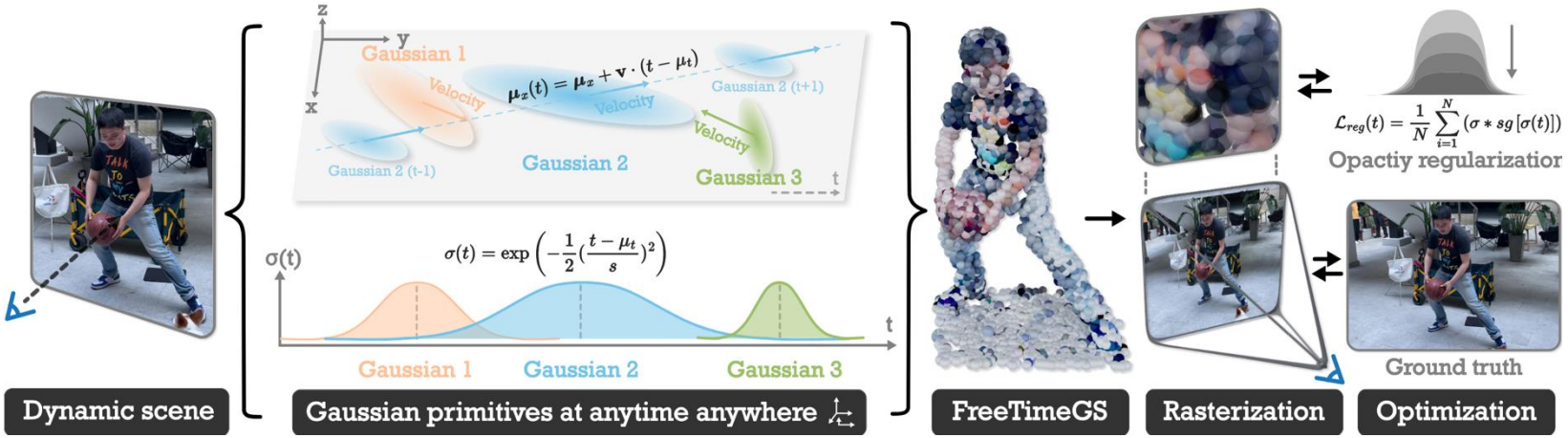
$\mu = \mathbf{\mu_x}$	$\Sigma = RSS^TR^T$	$\alpha = \sigma$	$c = \sum_{l=0}^{L}\sum_{m=-l}^{l} \mathbf{c}_{lm} Y_{lm}(\mathbf{d})$
위치	**크기와 방향**	**투명도**	**색상**

Background

- α-blending (alpha-blending) at Gaussian Primitive
  - 두 개 이상의 투명체를 렌더링하는 방법
    - 투명체는 α라는 투명도를 가짐
      - 카메라와 가까운 곳부터 투명도 혼합
      - 가우시안 렌더링 – alpha-blending 사용
        - ✓투명체들을 합치는 도중 $\alpha = 1$ (불투명함) 에 근사 시
          뒤에 추가적인 투명체(Gaussian Primitive)가 있어도 사용하지 않음
      - * 렌더링 : 3D Representation을 2D Image로 만드는 과정

Method Overview

1. Time-varying Gaussian Primitive
   - 위치, 투명도, 색상에 대해 시간 변수 추가
2. Opacity Regularization
   - 높은 Opacity로 인한 Gaussian Primitive 학습 방해 완화
3. Primitive Relocalization
   - Primitive가 너무 많아지는 것을 방지



Proposed Representation

- Time-varying Gaussian Primitive

$$G(\boldsymbol{x}, \mathrm{t}) = \alpha(t) \cdot \exp\left(-\frac{1}{2}(\boldsymbol{x} - \mu(\mathrm{t}))\Sigma^{-1}(\boldsymbol{x} - \mu(\mathrm{t}))\right)$$

$$\mu(t) = \mathbf{\mu_x} + \mathbf{v}(t - \mu_t)$$

위치

$$\Sigma = RSS^T R^T$$

크기와 방향

$$\alpha(\mathrm{t}) = \sigma \cdot \exp\left(-\frac{1}{2}\left(\frac{t - \mu_t}{s}\right)^2\right)$$

투명도

$$c = \sum_{l=0}^{L} \sum_{m=-l}^{l} \mathbf{c}_{lm} Y_{lm}\left(\mathbf{d}(\mu(t))\right)$$

색상

Proposed Representation

- Time-varying Gaussian Primitive

$$G(\boldsymbol{x}, t) = \alpha(t) \cdot \exp\left(-\frac{1}{2}(\boldsymbol{x} - \mu(t))\Sigma^{-1}(\boldsymbol{x} - \mu(t))\right)$$

위치 $\mu(t) = \boldsymbol{\mu}_{\mathbf{x}} + \mathbf{v}(t - \mu_t)$

- 속도 **v**를 통해 Gaussian Primitive를 직선 운동으로 모델링
- μ_t는 time offset

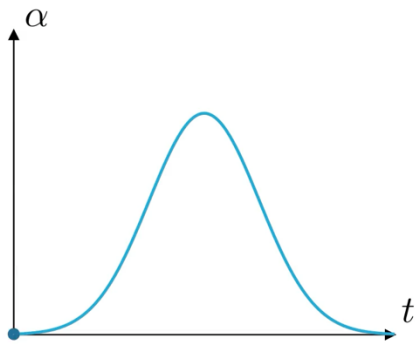
Proposed Representation

- Time-varying Gaussian Primitive

$$G(\boldsymbol{x}, t) = \alpha(t) \cdot \exp\left(-\frac{1}{2}(\boldsymbol{x} - \mu(t))\Sigma^{-1}(\boldsymbol{x} - \mu(t))\right)$$

투명도 $\alpha(t) = \sigma \cdot \exp\left(-\frac{1}{2}\left(\frac{t - \mu_t}{s}\right)^2\right)$

- 타임라인을 가우시안 함수로 설정
  - Fade In → 특정 시간에 투명도가 최대치 → Fade Out



Proposed Representation

- Time-varying Gaussian Primitive

$$G(\boldsymbol{x}, t) = \alpha(t) \cdot \exp\left(-\frac{1}{2}(\boldsymbol{x} - \mu(t))\Sigma^{-1}(\boldsymbol{x} - \mu(t))\right)$$

$$\mu(t) = \boldsymbol{\mu}_x + \mathbf{v}(t - \mu_t)$$

위치

$$\Sigma = RSS^T R^T$$

크기와 방향

$$\alpha(t) = \sigma \cdot \exp\left(-\frac{1}{2}\left(\frac{t - \mu_t}{s}\right)^2\right)$$

투명도

$$c = \sum_{l=0}^{L} \sum_{m=-l}^{l} \mathbf{c}_{lm} Y_{lm}\left(\mathbf{d}(\mu(t))\right)$$

색상

대상의 색은 관측 위치에 따라 다름 + 대상의 위치가 시간에 따라 변함
위치 그리고 시간에 따라 색이 다름

해당 Gaussian Primitive를 바라보는 방향벡터

Training

- Processing for Time-varying Gaussian Primitive
  - Dynamic scene을 만들기 위해 저자는 다음과 같은 Process를 진행
    - Initialization – Feature Matching(RoMa) + MVS
      - Camera Pose가 있는 이미지들을 Input
    - Loss – 4D Regularization (Opacity Regularization)
    - Iterative Process – Primitive Relocation

Training

- Processing for Time-varying Gaussian Primitive – Loss
  - Rendering Loss

$$\mathcal{L}_{\text{render}} = \mathcal{L}_{\text{render}} + \mathcal{L}_{\text{PSNR}} + \mathcal{L}_{\text{SSIM}}$$

  - 4D Regularization (Opacity Regularization)
    - Rendering Loss만 사용시, 빠르거나 복잡한 Motion에서 렌더링 성능이 떨어짐
    - Opacity 분포가 1.0 으로 몰린 것을 확인

      ∴ alpha-blending 특징에 의해 많은 가우시안들이 Backpropagation 할 수 없어짐

$$\mathcal{L}_{reg}(t) = \frac{1}{N}\sum_{i=1}^{N}(\sigma \cdot sg[\sigma(t)])$$

Training

- Processing for Time-varying Gaussian Primitive – Iterative Process
  - Primitive Relocation
    - Opacity Regularization을 통해 렌더링 성능을 높아짐
    - BUT Gaussian Primitive 수가 많아짐
    - N iteration 마다 Primitive Relocation 진행
      - $\mathbf{s} = \lambda_g \nabla_g + \lambda_o \sigma$
      - α가 작을 때, sampling score $\mathbf{s}$가 높은 곳으로 이동시킴

Experiments

- Experiments
  - 저자가 주장하는 모든 모듈이 있어야 성능 잘 나옴

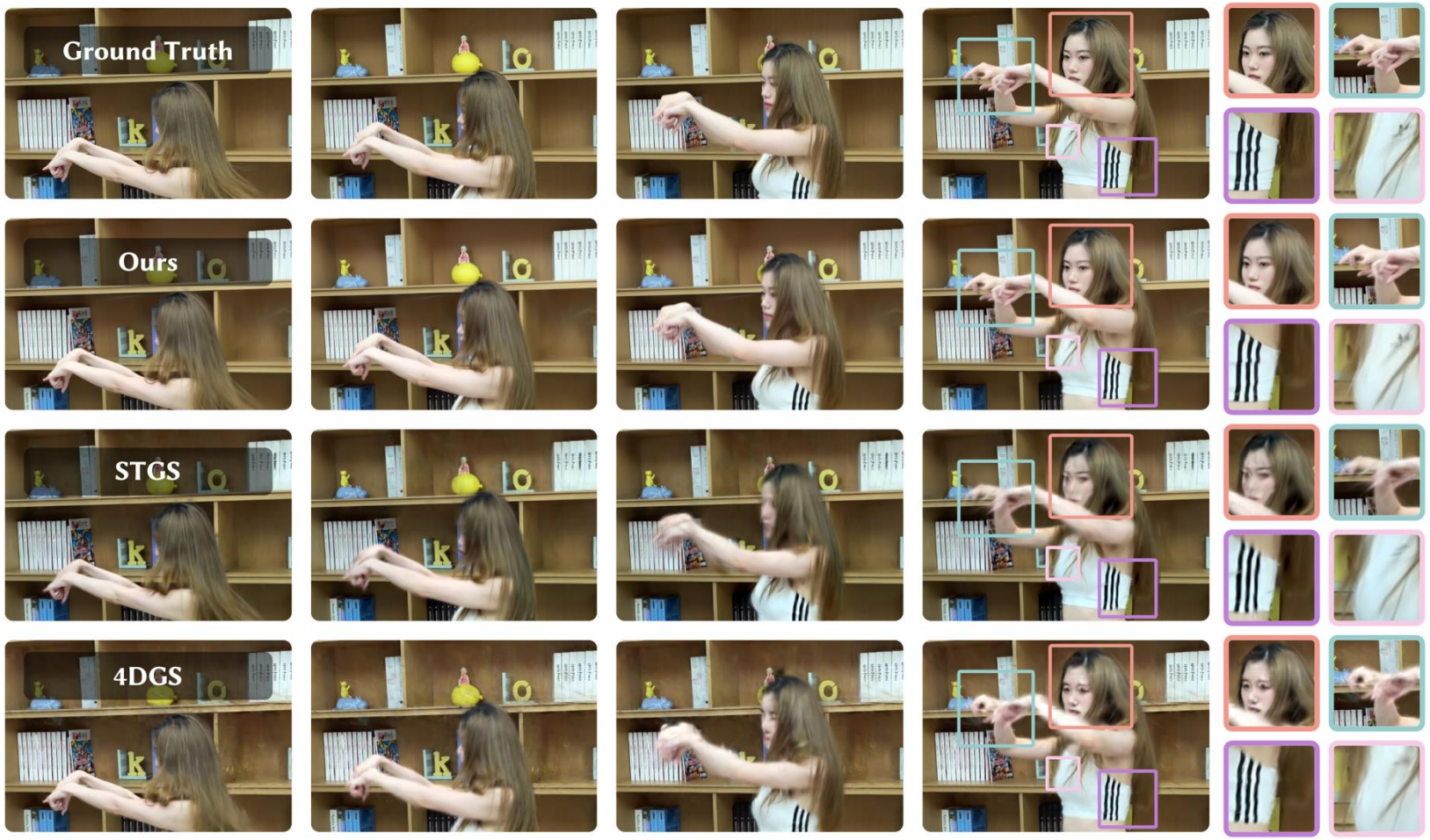


Ground Truth | Ours | *w/o* our motion | *w/o* 4d regularization | *w/o* periodic relocation | *w/o* 4d initialization

	PSNR↑	DSSIM$_2$↓	LPIPS↓
w/o our motion	28.10/26.92	0.024/0.031	0.165/0.161
w/o 4d regularization	28.68/29.09	0.023/0.024	0.159/0.150
w/o periodic relocation	29.07/29.15	0.020/0.021	0.155/0.146
w/o 4d initialization	28.33/27.06	0.023/0.030	0.162/0.158
Ours	29.74/30.75	0.018/0.017	0.152/0.133

Experiments

- Experiments – SelfCap
  - 저자들이 직접 구축한 SelfCap 데이터셋



Experiments

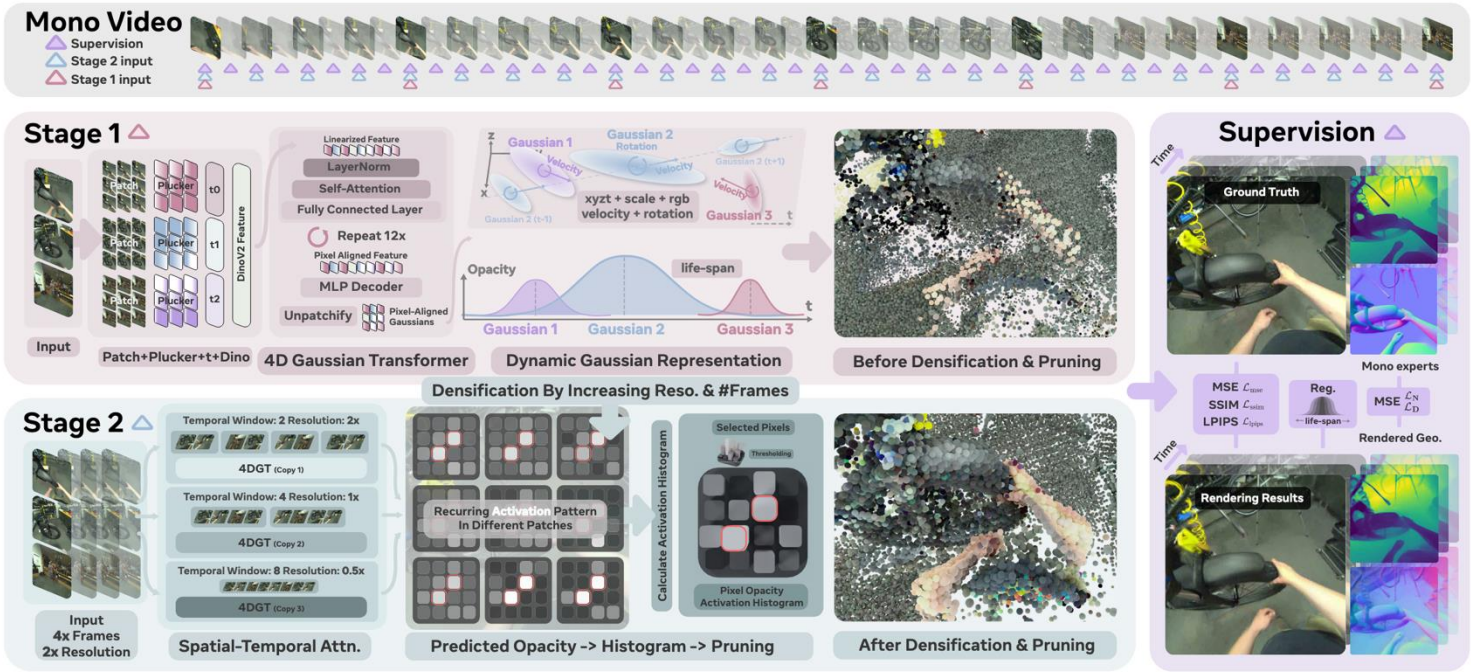
- Experiments – SelfCap
  - 저자들이 직접 구축한 SelfCap 데이터셋에서 성능 비교
    - 4K 60fps. 22 Camera
    - 메모리 대비 정량지표과 주사율 모두 뛰어나다

	PSNR↑	$DSSIM_2$↓	LPIPS↓	FPS↑	Size (MB) ↓
Deformable-3DGS [44]	25.95/25.27	0.037/0.026	0.298/0.139	57	73
STGS [21]	24.97/25.32	0.048/0.029	0.273/0.123	142	77
4DGS [49]	25.98/26.75	0.036/0.019	0.237/0.104	65	827
Ours	27.41/29.38	0.024/0.013	0.204/0.080	467	96
Ours†	27.27/28.87	0.025/0.013	0.217/0.081	664	53

4DGT:
Learning a 4D Gaussian Transformer Using Real-World Monocular Videos [NeurIPS 2025 Spotlight]

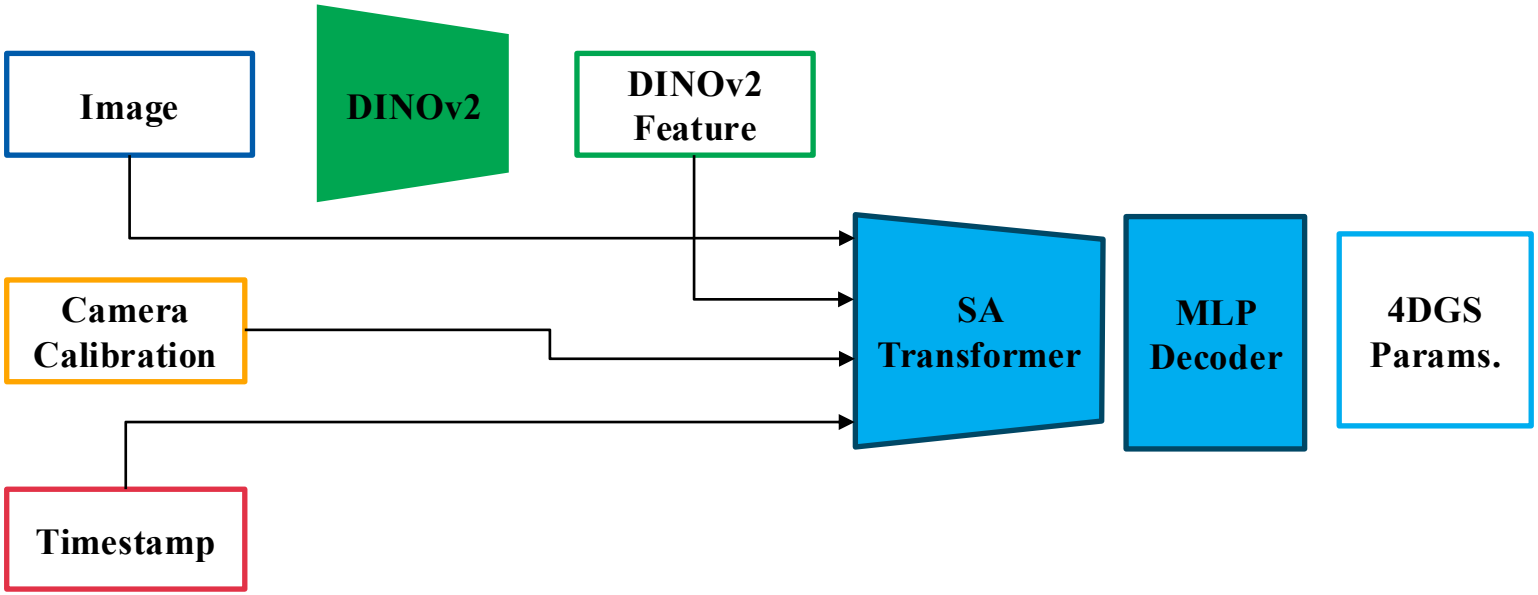
Method Overview

- Feed Forward Network at 4DGS
  - 4D Gaussian을 Optimization 하지 않고 Inference를 수행
- Unifies Static and Dynamic Components
  - Lifespan(수명)이라는 개념을 통해 정적/동적 요소를 한 모델에서 통합
- Efficient Pruning and Densify Strategy
  - 새로운 Pruning, Densify 기법 제안



Methods

- Feed-Forward Dynamic Gaussian Prediction
  - Image, Camera Calibration, Timestamp, DINOv2 Feature를 Concat한 것을 입력으로 받음.
  - Self-Attention Transformer를 통과해 Feature Fusion
  - 최종적으로 MLP Decoder를 통해 Time-varying 2D Gaussian의 파라미터를 출력함



Methods

- Time-varying Gaussian Primitive
  - 2D Gaussian을 사용
  - 회전에 대해서 시간 변수 추가

$$G(\boldsymbol{x}, \mathrm{t}) = \alpha(t) \cdot \exp\left(-\frac{1}{2}(\boldsymbol{x} - \mu(\mathrm{t}))\Sigma(\mathrm{t})^{-1}(\boldsymbol{x} - \mu(\mathrm{t}))\right)$$

  - $s \to \infty$이며 위치와 방향이 거의 안 변하면 정적 장면으로 취급

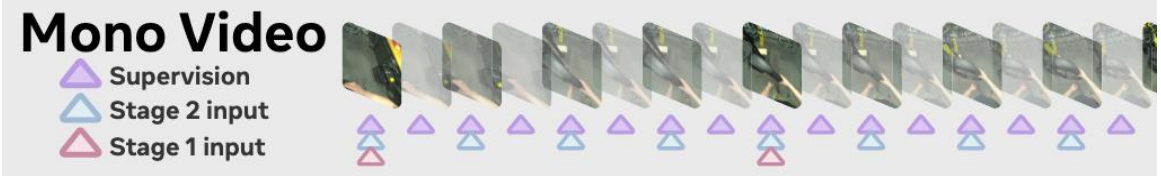
$$\alpha(\mathrm{t}) = \sigma \cdot \exp\left(-\frac{1}{2}\left(\frac{t - \mu_t}{s}\right)^2\right)$$

Methods

- Loss

$$\mathcal{L} = \mathcal{L}_{\text{mse}} + \lambda_{\text{lpips}}\mathcal{L}_{\text{lpips}} + \lambda_{\text{ssim}}\mathcal{L}_{\text{ssim}} + \lambda_{\mathbf{v}}\mathcal{L}_{\mathbf{v}} + \lambda_{\boldsymbol{\omega}}\mathcal{L}_{\boldsymbol{\omega}} + \lambda_{\mathbf{l}}\mathcal{L}_{\mathbf{l}} + \lambda_{\mathbf{D}}\mathcal{L}_{\mathbf{D}} + \lambda_{\mathbf{N}}\mathcal{L}_{\mathbf{N}}$$

Self-Supervise Loss | **Regularization Loss** | **Expert Loss**



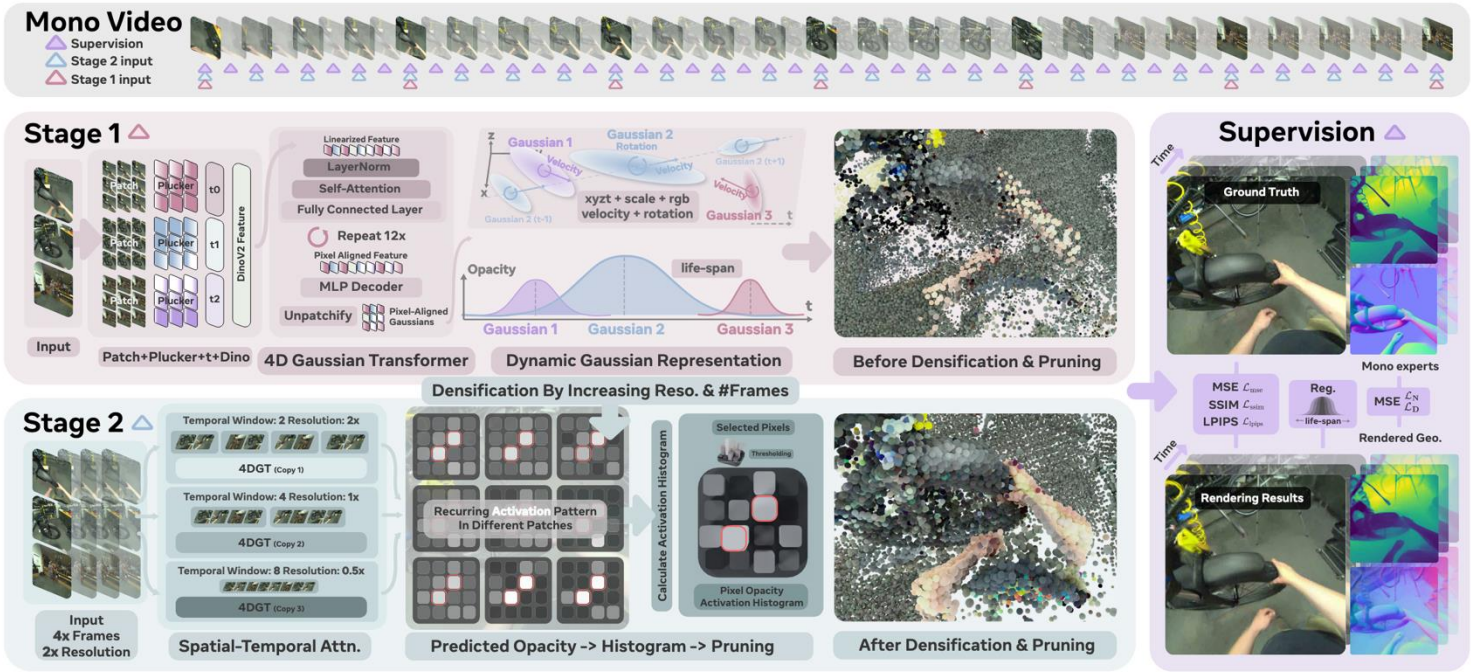
$$\mathcal{L}_{\text{mse}} = \sum_{i=1}^{W} \frac{\left\| \mathbf{I}_i - \mathbf{I}_i^{'} \right\|_2}{W}, \quad \mathcal{L}_{\text{lpips}} = \sum_{i=1}^{W} \frac{\left\| \psi(\mathbf{I}_i) - \psi(\mathbf{I}_i^{'}) \right\|_1}{W}, \quad \mathcal{L}_{\text{ssim}} = \sum_{i=1}^{W} \frac{\text{SSIM}(\mathbf{I}_i, \mathbf{I}_i^{'})}{W}$$

$$\mathcal{L}_{\mathbf{v}} = \sum_{i=1,j=1,k=1}^{N,HW/p^2,p^2} \frac{\left\| \mathbf{v}_{i,j,k} \right\|_1}{NHW}, \quad \mathcal{L}_{\boldsymbol{\omega}} = \sum_{i=1,j=1,k=1}^{N,HW/p^2,p^2} \frac{\left\| \boldsymbol{\omega}_{i,j,k} \right\|_1}{NHW}, \quad \mathcal{L}_{\mathbf{l}} = \sum_{i=1,j=1,k=1}^{N,HW/p^2,p^2} \frac{\left\| \frac{1}{\mathbf{l}_{i,j,k}} \right\|_1}{NHW}$$

$$\mathcal{L}_{\mathbf{D}} = \sum_{i=1}^{W} \frac{\left\| \mathbf{D}_i - \mathbf{D}_i^{'} \right\|_2}{W}, \quad \mathcal{L}_{\mathbf{N}} = \sum_{i=1}^{W} \frac{\left\| \mathbf{N}_i - \mathbf{N}_i^{'} \right\|_2}{W}$$

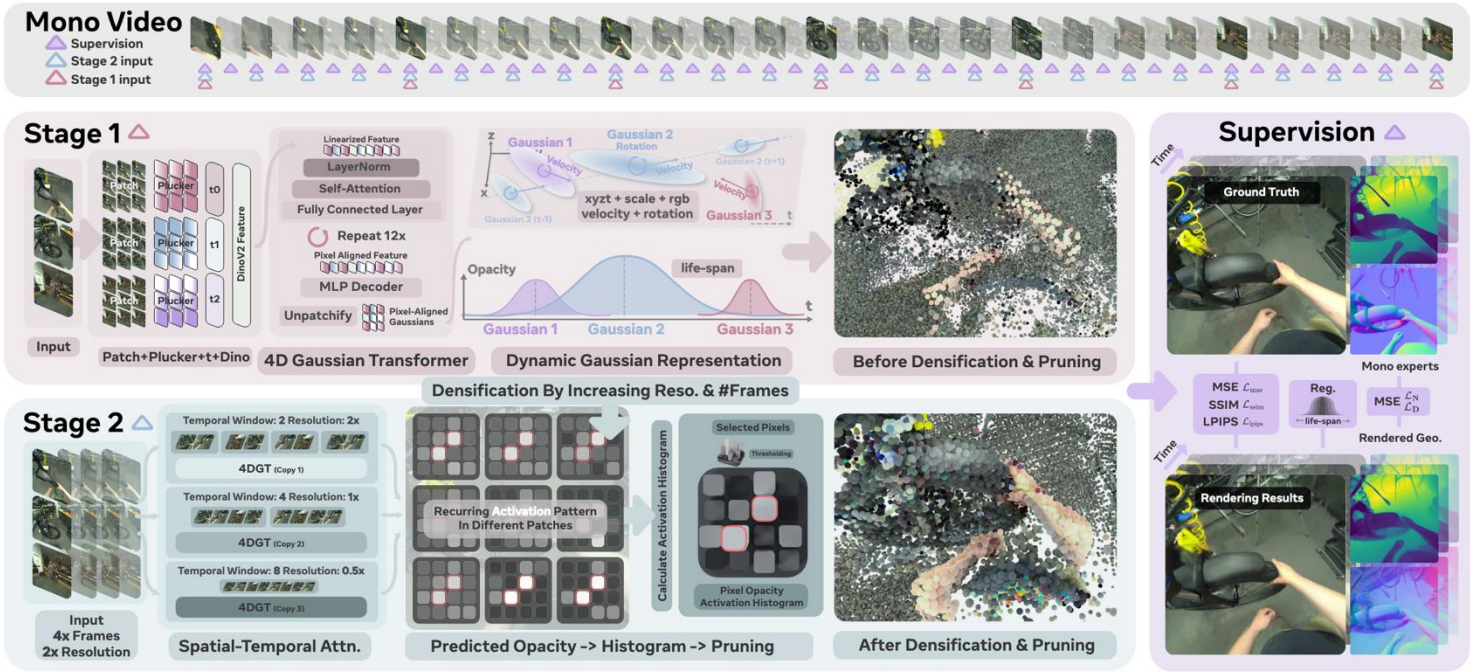
Methods

- Pruning and Densify Strategy
  - Pruning
    - Opacity를 정규화했을 때 기준 Z score가 1.0 초과하는 토큰의 각 위치에서의 빈도수 측정
    - 빈도가 높은 상위 토큰 S=10 개 선택
      - 매우 큰폭으로 줄임



Methods

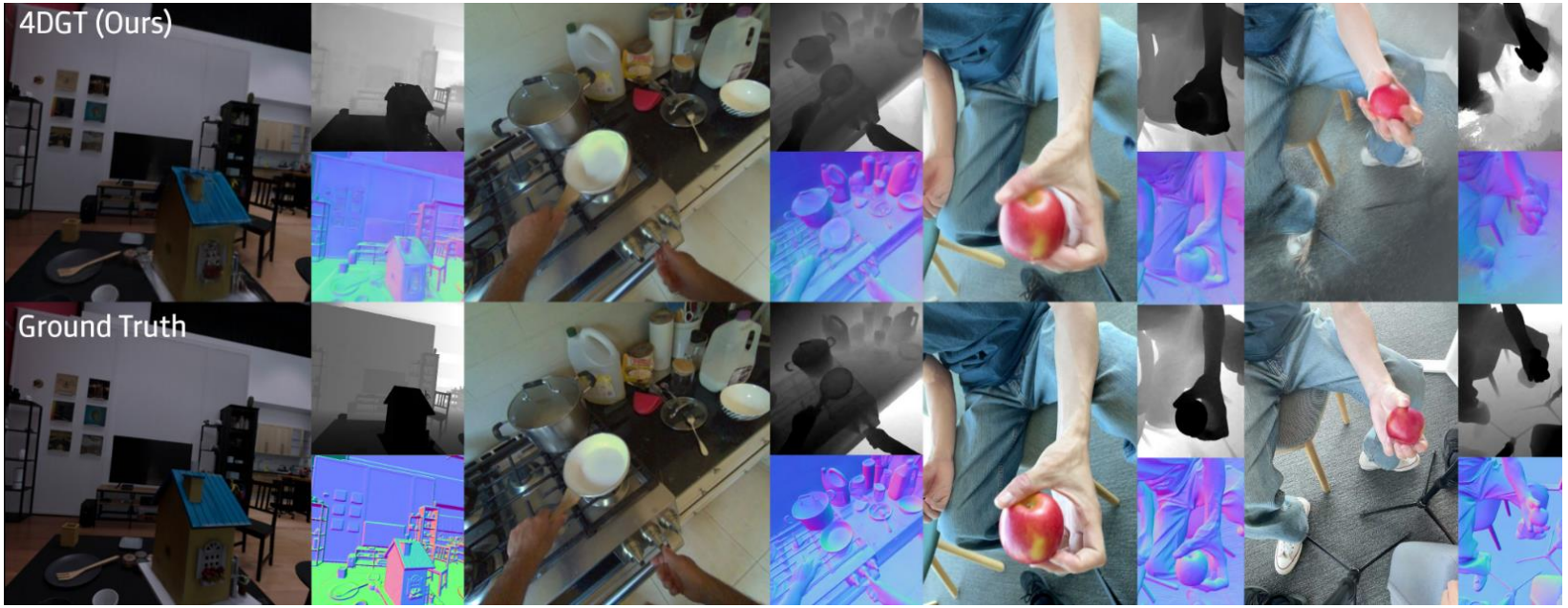
- Pruning and Densify Strategy
  - Densification
    - Spatial하게 Resolution 올림
    - Temporal하게 frame수 올림



Experiments

- Experiments
  - Train at 64xH100 9 + 6 days
  - Inference at A100 80G
  - 매우 빠른 시간

Method	PSNR (Render) ↑ ADT	TUM	DyC	Aria	Avg	LPIPS (Render) ↓ ADT	TUM	DyC	Aria	Avg	RMSE (Depth) ↓ ADT	TUM	Avg	Deg. ↓ ADT	Recon. Time ↓
SoM [53]	±1.640 30.30	±0.266 21.03	±2.597 16.49	±3.701 26.69	**23.63**	±0.024 0.242	±0.069 0.337	±0.060 0.392	±0.067 0.307	**0.320**	±3.000 4.158	±1.354 2.756	±1.633 **3.434**	±3.607 35.05	60000 ms / f
L4GM [43]	±2.713 7.348	±0.112 9.226	±0.629 8.770	±1.666 8.617	**8.490**	±0.051 0.688	±0.013 0.670	±0.062 0.587	±0.052 0.698	**0.661**	±0.660 2.606	±0.791 1.698	±0.386 **2.094**	±4.181 63.51	200 ms / f
MonST3R [68, 66]	±1.651 25.13	±1.673 20.61	±1.673 11.32	±3.355 19.90	**19.24**	±0.011 0.246	±0.023 0.273	±0.045 0.429	±0.135 0.323	**0.318**	±0.842 2.111	±0.243 0.653	±0.543 **1.382**	±2.699 25.00	4500 ms / f
Experts [40, 66]	±3.195 23.32	±3.117 18.64	±1.258 11.53	±2.419 22.32	**18.96**	±0.082 0.299	±0.073 0.318	±0.058 0.423	±0.046 0.236	**0.319**	±0.621 2.931	±0.074 0.919	±0.348 **1.925**	±0.748 26.26	350 ms / f
Ours	±1.508 28.31	±0.048 21.02	±2.034 16.12	±1.591 27.36	**23.20**	±0.021 0.243	±0.009 0.349	±0.067 0.408	±0.019 0.230	**0.308**	±0.463 0.934	±0.048 0.394	±0.255 **0.664**	±1.831 25.92	25 ms / f



Future Works

- 연구주제 잡기
  - 4DGS로 더 따라가보기
  - ETRI 과제