3D Gaussian Splatting
&
NeRF

BrainLab of Seoul Tech National University

Department of Applied Artificial Intelligence

Jeong Sang-Yeop

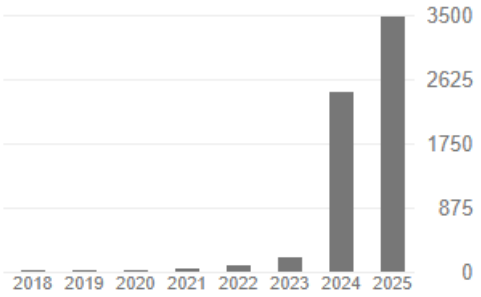
BrAIn Lab.
Brain and Artificial Intelligence Lab.

I. Author

Bernhard Kerbl

TU Wien
cg.tuwien.ac.at의 이메일 확인됨 - 홈페이지
GPU real-time graphics parallel processing

인용		모두 보기
	전체	2020년 이후
서지정보	6524	6367
h-index	16	16
i10-index	22	21



3D Gaussian Splatting for Real-time Radiance Field Rendering B Kerbl, G Kopanas, T Leimkühler, G Drettakis ACM Transactions on Graphics (ToG) 42 (4), 1-14	5377	2023
A Hierarchical 3D Gaussian Representation for Real-Time Rendering of Very Large Datasets B Kerbl, A Meuleman, G Kopanas, M Wimmer, A Lanvin, G Drettakis ACM Transactions on Graphics 43 (4)	160	2024
Reducing the memory footprint of 3d gaussian splatting P Papantonakis, G Kopanas, B Kerbl, A Lanvin, G Drettakis Proceedings of the ACM on Computer Graphics and Interactive Techniques 7 (1 ...	107	2024
Retargeting technical documentation to augmented reality P Mohr, B Kerbl, M Donoser, D Schmalstieg, D Kalkofen Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing ...	102	2015
Nerfshop: Interactive editing of neural radiance fields C Jambon, B Kerbl, G Kopanas, S Diolatzis, T Leimkühler, G Drettakis Proceedings of the ACM on Computer Graphics and Interactive Techniques 6 (1)	92	2023
Taming 3DGS: High-Quality Radiance Fields with Limited Resources S Subhajyoti Mallick, R Goel, B Kerbl, FV Carrasco, M Steinberger, ... Conference Proceedings of SIGGRAPH Asia	87 *	2024
Whippletree: Task-based scheduling of dynamic workloads on the GPU M Steinberger, M Kenzel, P Boechat, B Kerbl, M Dokter, D Schmalstieg ACM Transactions on Graphics (TOG) 33 (6), 1-11	70	2014
Stopthepop: Sorted gaussian splatting for view-consistent real-time rendering L Radl, M Steiner, M Parger, A Weinrauch, B Kerbl, M Steinberger ACM Transactions on Graphics (TOG) 43 (4), 1-17	69	2024
Rendering Point Clouds with Compute Shaders and Vertex Order Optimization M Schütz, B Kerbl, M Wimmer Computer Graphics Forum 40 (4), 115-126	67	2021
Softshell: dynamic scheduling on GPUs M Steinberger, B Kainz, B Kerbl, S Hauswiesner, M Kenzel, ... ACM Transactions on Graphics (TOG) 31 (6), 1-11	61	2012
Software Rasterization of 2 Billion Points in Real Time M Schütz, B Kerbl, M Wimmer Proceedings of the ACM on Computer Graphics and Interactive Techniques 5 (3)	44	2022
A high-performance software graphics pipeline architecture for the GPU M Kenzel, B Kerbl, D Schmalstieg, M Steinberger ACM Transactions on Graphics (TOG) 37 (4), 1-15	38	2018

II. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis

II. NeRF - Overview

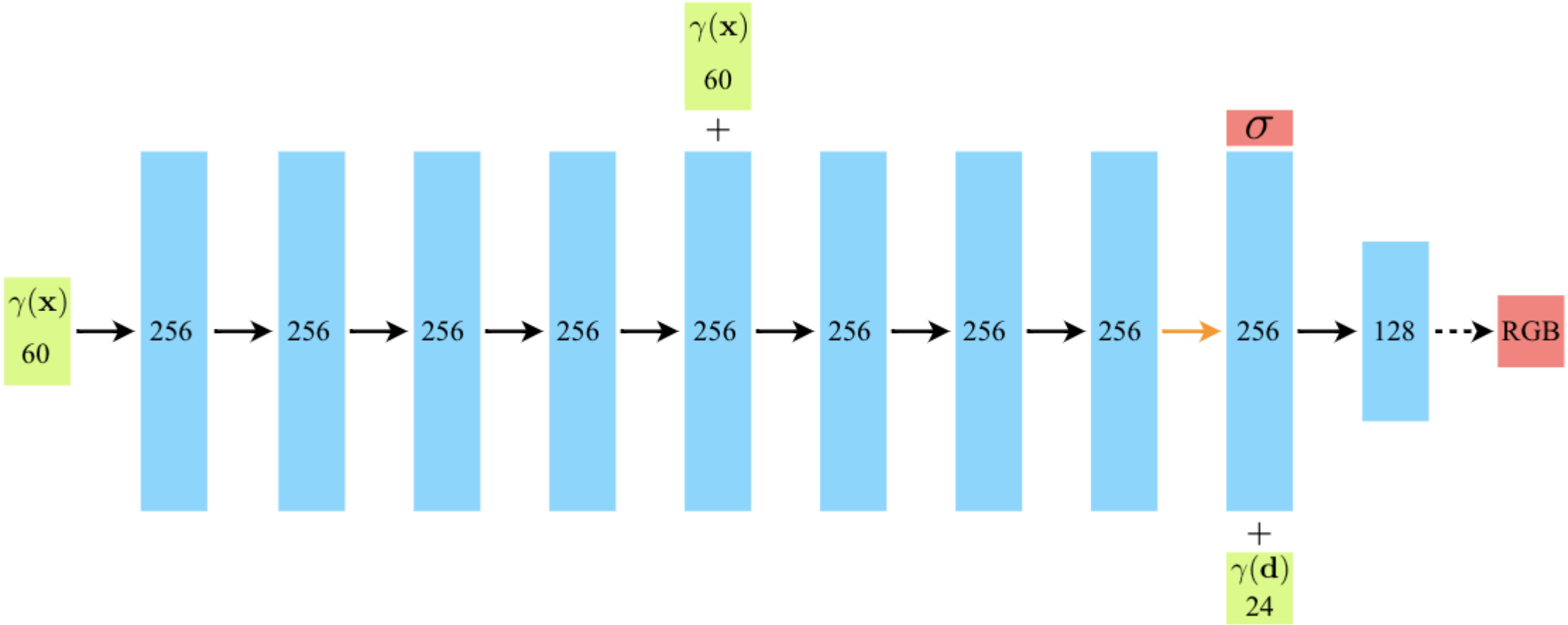
BrAIn Lab.
Brain and Artificial Intelligence Lab.

"현실 이미지를 다양한 카메라
각도에서 High Quality로
Rendering하는 방법"

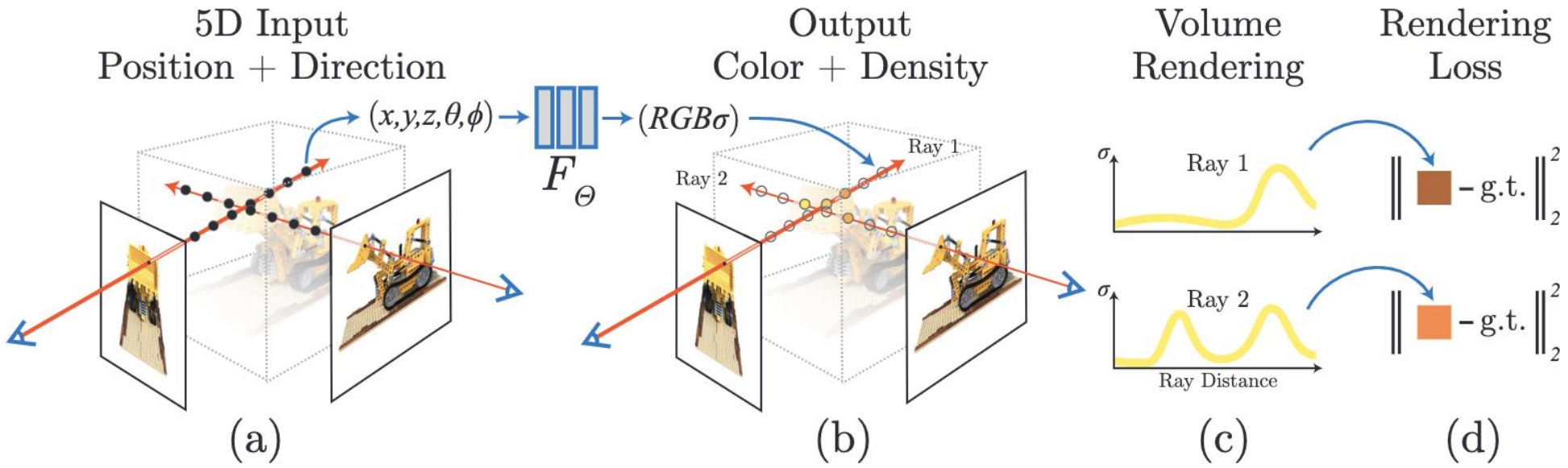
II. NeRF - Overview

II. NeRF - Process

BrAIn Lab.
Brain and Artificial Intelligence Lab.



II. NeRF – Pixel Color Function



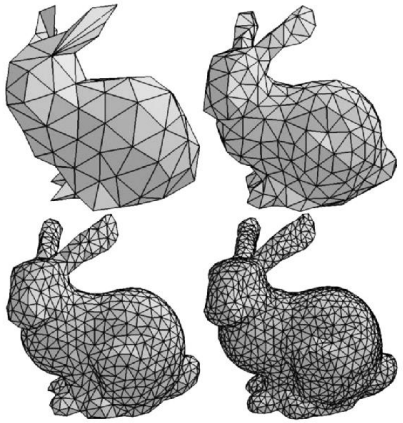
$$r(t) = o + td$$

$$C(r) = \int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t), d)\, dt, \qquad \text{where } T(t) = \exp\left(-\int_{t_n}^{t_f} \sigma(r(s))ds\right)$$

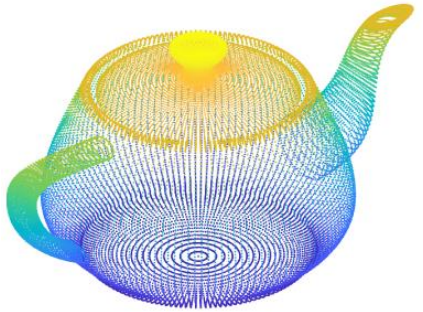
III. 3D Gaussian splatting for real-time radiance field rendering

III. 3DGS – Needs

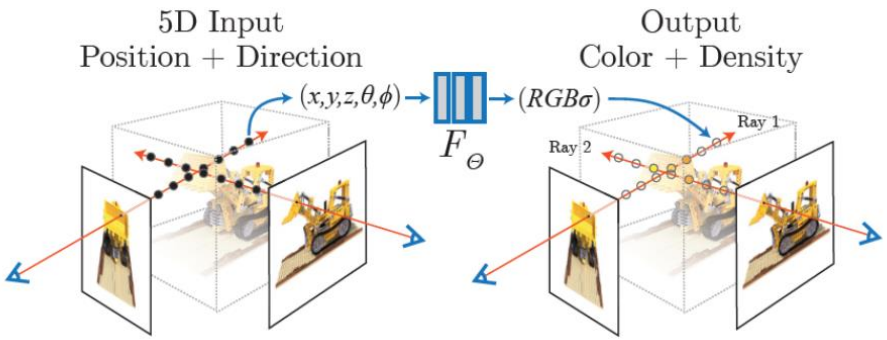
<Voxel>



<Mesh>

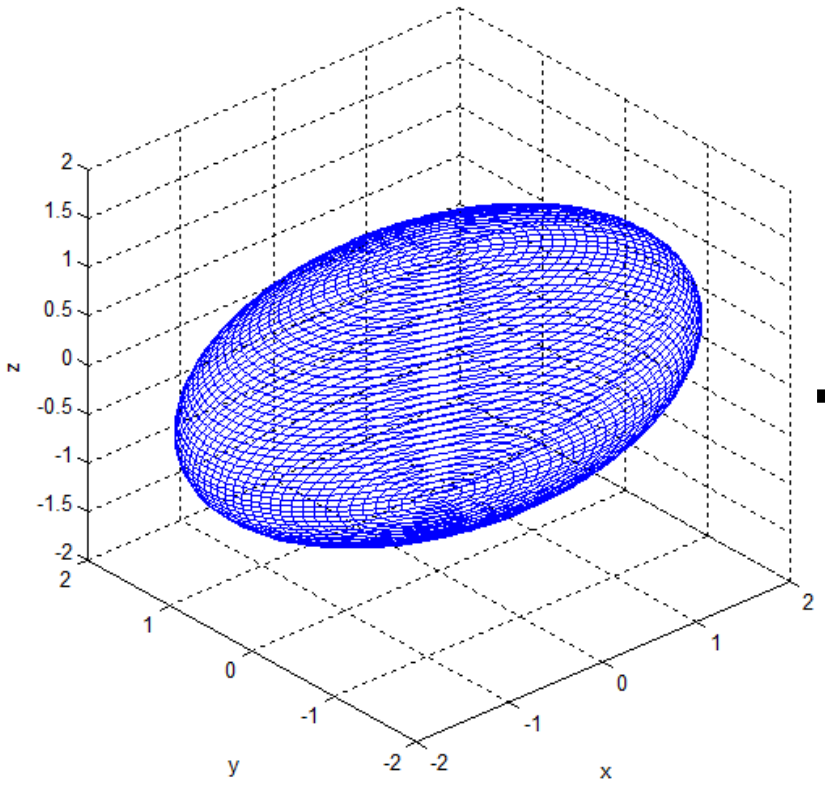


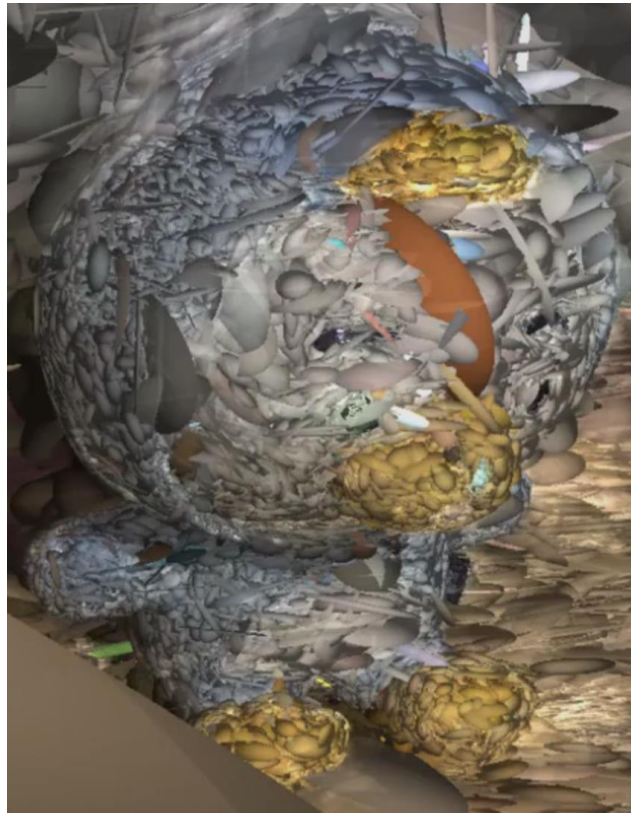
<Point Cloud>



<NeRF>

III. 3DGS – Point 대신 3d Gaussian으로



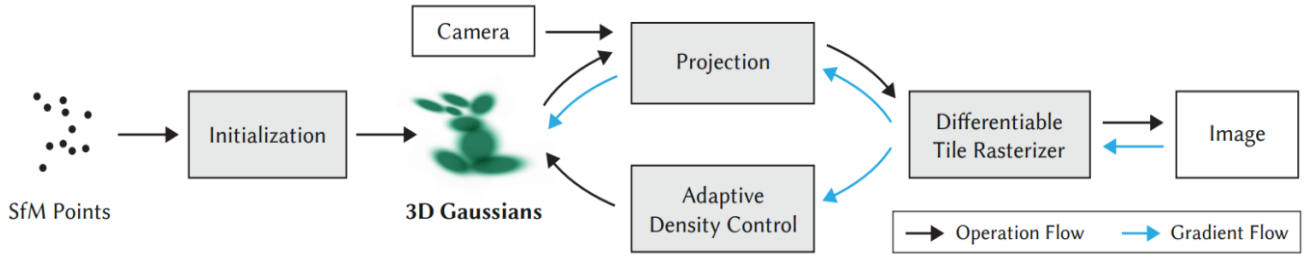


III. 3DGS – Process

BrAIn Lab.
Brain and Artificial Intelligence Lab.

Algorithm 1 Optimization and Densification
w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while



III. 3DGS – Process(Initialization)

BrAIn Lab.
Brain and Artificial Intelligence Lab.

Algorithm 1 Optimization and Densification

w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

mean (SfM)

covariance

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

color (SH coefficients)

opacity (0~1 sigmoid)

III. 3DGS – Process(Initialization)

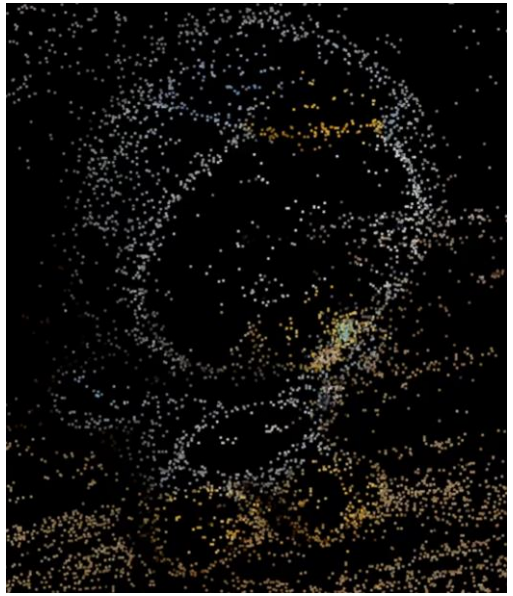
BrAIn Lab.
Brain and Artificial Intelligence Lab.

Algorithm 1 Optimization and Densification

w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

SfM(Structure from Motion)



III. 3DGS – Process(Initialization)

BrAIn Lab. Brain and Artificial Intelligence Lab.

Algorithm 1 Optimization and Densification

w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

Covariance

$$\sum = RSIS^TR^T$$

$$Cov(X,X) = E[(X - E[X])(X - E[X])^T] = E[XX^T] - E[X]E[X^T]$$

$$Cov(X,X) = \begin{bmatrix} Cov(X_1,X_1) & Cov(X_1,X_2) & \cdots & Cov(X_1,X_n) \\ Cov(X_2,X_1) & Cov(X_2,X_2) & \cdots & Cov(X_2,X_n) \\ \vdots & \vdots & \ddots & \vdots \\ Cov(X_n,X_1) & Cov(X_n,X_2) & \cdots & Cov(X_n,X_n) \end{bmatrix}$$

$$Cov(X,X) = \begin{bmatrix} Var(X_1) & Cov(X_1,X_2) & \cdots & Cov(X_1,X_n) \\ Cov(X_2,X_1) & Var(X_2) & \cdots & Cov(X_2,X_n) \\ \vdots & \vdots & \ddots & \vdots \\ Cov(X_n,X_1) & Cov(X_n,X_2) & \cdots & Var(X_n) \end{bmatrix}$$

III. 3DGS – Process(Initialization)

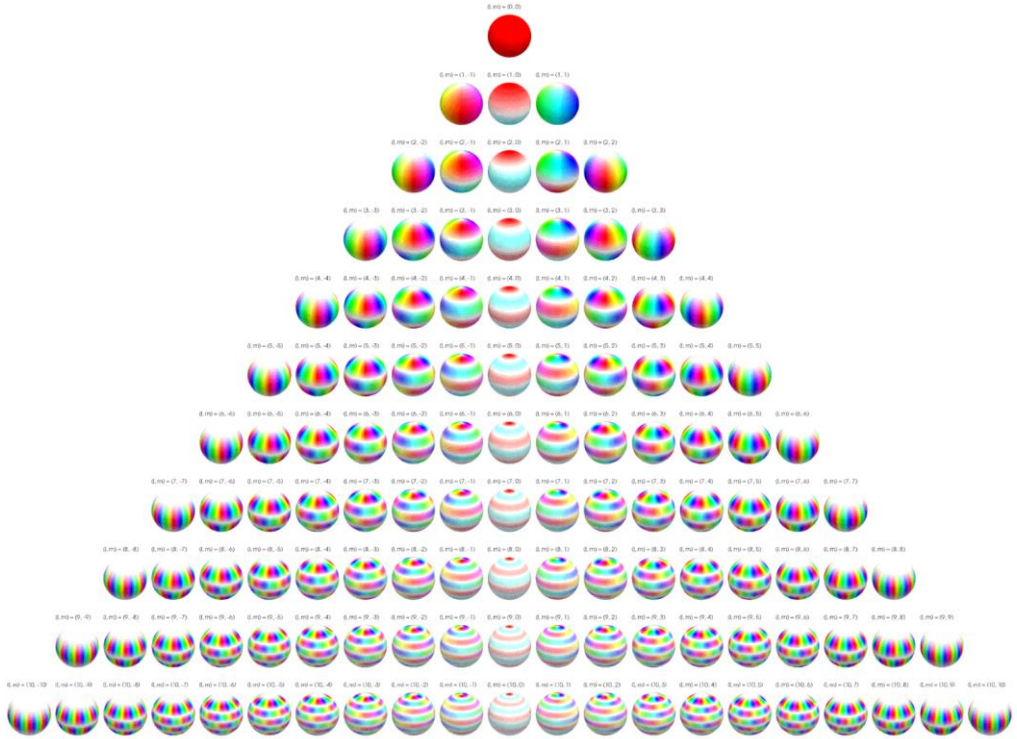
BrAIn Lab.
Brain and Artificial Intelligence Lab.

Algorithm 1 Optimization and Densification

w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

SH(Spherical Harmonics) coefficients



III. 3DGS – Process(Projection)

BrAIn Lab.
Brain and Artificial Intelligence Lab.

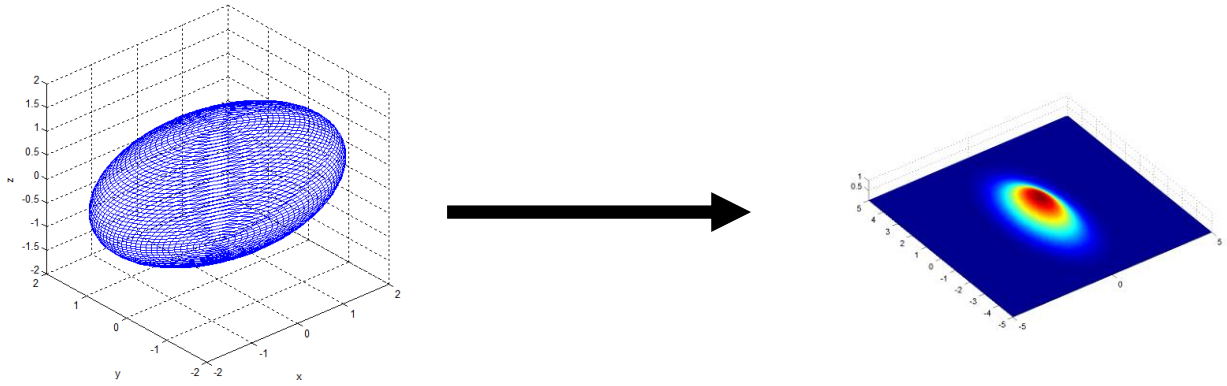
Algorithm 1 Optimization and Densification
w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

이미지 좌표계에서의 Covariance

$$\Sigma' = \overset{\text{Matrix}}{JW\Sigma W^T J^T}$$

- W: 3D 공간에 흩어져 있는 모든 객체들을 특정 카메라의 시점에서 본 모습으로 재정렬하는 Matrix
- J: Camera 좌표계에서 Image 좌표계로 변환하는 Matrix(Jacobian)



III. 3DGS – Process(Projection)

BrAIn Lab.
Brain and Artificial Intelligence Lab.

Algorithm 1 Optimization and Densification
w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

III. 3DGS – Process(Optimization)

BrAIn Lab. Brain and Artificial Intelligence Lab.

Algorithm 1 Optimization and Densification
w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_1 + \lambda\mathcal{L}_{\text{D-SSIM}}$$

$$\lambda = 0.2$$

$$\text{SSIM}(\mathbf{x}, \mathbf{y}) = [l(\mathbf{x}, \mathbf{y})]^{\alpha} \cdot [c(\mathbf{x}, \mathbf{y})]^{\beta} \cdot [s(\mathbf{x}, \mathbf{y})]^{\gamma}$$

↓

Structural Similarity Index Measure

Luminance(휘도), Constrast(대비), Structure(구조)

↓

$$L_{D-SSIM} = (1 - SSIM) / 2$$

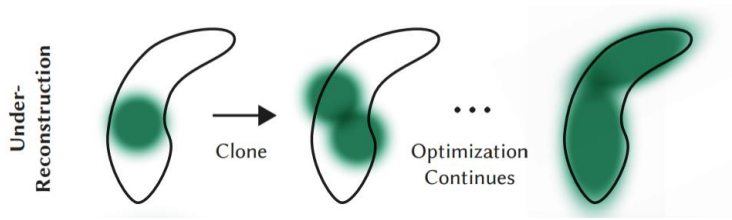
III. 3DGS – Process(Adaptive Control of Gaussians)

Algorithm 1 Optimization and Densification

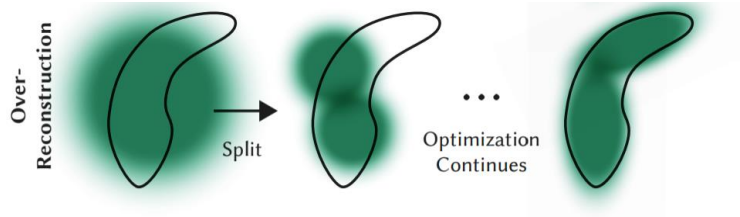
w, h: width and height of the training images

$M \leftarrow$ SfM Points ▷ Positions
$S, C, A \leftarrow$ InitAttributes() ▷ Covariances, Colors, Opacities
$i \leftarrow 0$ ▷ Iteration Count
while not converged **do**
  $V, \hat{I} \leftarrow$ SampleTrainingView() ▷ Camera V and Image
  $I \leftarrow$ Rasterize(M, S, C, A, V) ▷ Alg. 2
  $L \leftarrow Loss(I, \hat{I})$ ▷ Loss
  $M, S, C, A \leftarrow$ Adam(∇L) ▷ Backprop & Step
  if IsRefinementIteration(i) **then**
    for all Gaussians (μ, Σ, c, α) **in** (M, S, C, A) **do**
      if $\alpha < \epsilon$ or IsTooLarge(μ, Σ) **then** ▷ Pruning
        RemoveGaussian()
      end if
      if $\nabla_p L > \tau_p$ **then** ▷ Densification
        if $\|S\| > \tau_S$ **then** ▷ Over-reconstruction
          SplitGaussian(μ, Σ, c, α)
        else ▷ Under-reconstruction
          CloneGaussian(μ, Σ, c, α)
        end if
      end if
    end for
  end if
  $i \leftarrow i + 1$
end while

Gradient가 너무 크다!



Following the gradient of position



Sampling from the parent PDF + 1.6 scaling

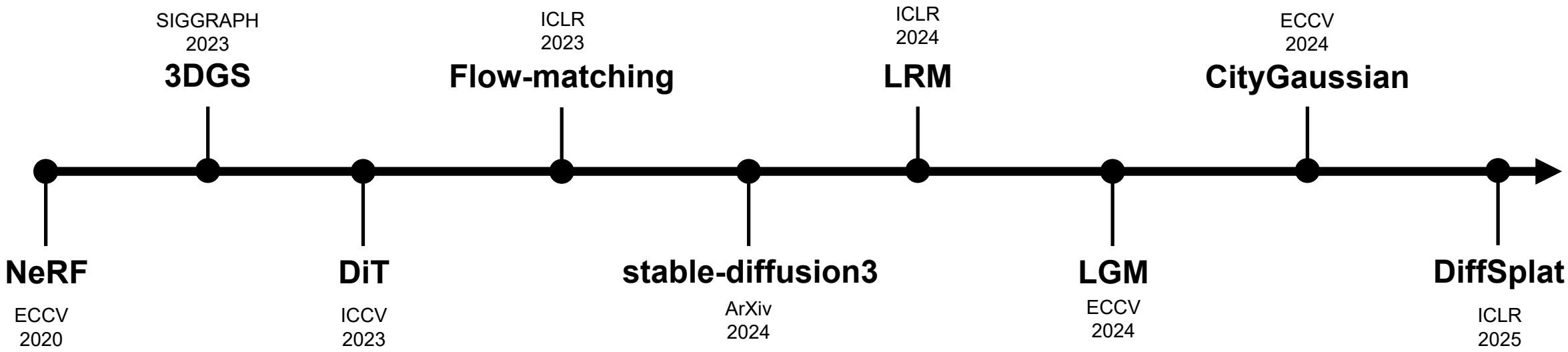
III. 3DGS – Result

Dataset	Mip-NeRF360						Tanks&Temples						Deep Blending					
Method|Metric	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem	*SSIM*↑	*PSNR*↑	*LPIPS*↓	Train	FPS	Mem
Plenoxels	0.626	23.08	0.463	25m49s	6.79	2.1GB	0.719	21.08	0.379	25m5s	13.0	2.3GB	0.795	23.06	0.510	27m49s	11.2	2.7GB
INGP-Base	0.671	25.30	0.371	5m37s	11.7	13MB	0.723	21.72	0.330	5m26s	17.1	13MB	0.797	23.62	0.423	6m31s	3.26	13MB
INGP-Big	0.699	25.59	0.331	7m30s	9.43	48MB	0.745	21.92	0.305	6m59s	14.4	48MB	0.817	24.96	0.390	8m	2.79	48MB
M-NeRF360	0.792†	27.69†	0.237†	48h	0.06	8.6MB	0.759	22.22	0.257	48h	0.14	8.6MB	0.901	29.40	0.245	48h	0.09	8.6MB
Ours-7K	0.770	25.60	0.279	6m25s	160	523MB	0.767	21.20	0.280	6m55s	197	270MB	0.875	27.78	0.317	4m35s	172	386MB
Ours-30K	0.815	27.21	0.214	41m33s	134	734MB	0.841	23.14	0.183	26m54s	154	411MB	0.903	29.41	0.243	36m2s	137	676MB

IV. Recent Advances

IV. Recent Advances

BrAIn Lab.
Brain and Artificial Intelligence Lab.



IV. Recent Advances(DiT)

Scalable Diffusion Models with Transformers

William Peebles* UC Berkeley
Saining Xie New York University

Figure 1: **Diffusion models with transformer backbones achieve state-of-the-art image quality.** We show selected samples from two of our class-conditional DiT-XL/2 models trained on ImageNet at 512×512 and 256×256 resolution.

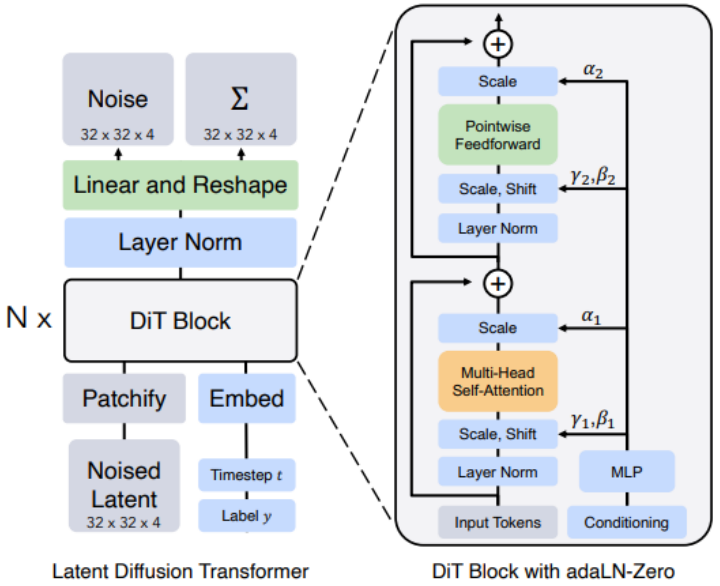
Abstract

We explore a new class of diffusion models based on the transformer architecture. We train latent diffusion models of images, replacing the commonly-used U-Net backbone with a transformer that operates on latent patches. We analyze the scalability of our Diffusion Transformers (DiTs) through the lens of forward pass complexity as measured by Gflops. We find that DiTs with higher Gflops—through increased transformer depth/width or increased number of input tokens—consistently have lower FID. In addition to possessing good scalability properties, our largest DiT-XL/2 models outperform all prior diffusion models on the class-conditional ImageNet 512×512 and 256×256 benchmarks, achieving a state-of-the-art FID of 2.27 on the latter.

1. Introduction

Machine learning is experiencing a renaissance powered by transformers. Over the past five years, neural architectures for natural language processing [42, 8], vision [10] and several other domains have been subsumed by transformers [60]. Many classes of image-level generative models remain holdouts to the trend, though—while transformers see widespread use in autoregressive models [43, 3, 6, 47], they have seen less adoption in other generative modeling frameworks. For example, diffusion models have been at the forefront of recent advances in image generation [9, 46]; yet, they all adopt a convolutional U-Net architecture as the de-facto choice of backbone.

* Work done during an internship at Meta AI, FAIR Team. Code and project page available here.



Latent Diffusion Transformer — DiT Block with adaLN-Zero

1. 확산 모델의 표준이었던 U-Net 아키텍처 대신, Vision Transformer 의 디자인을 계승한 Transformer 기반의 Diffusion Transformer를 제안.
2. 모델의 계산 복잡도를 증가시킬수록 생성 품질이 예측 가능하게 향상되는 DiT의 강력한 확장성을 입증.

IV. Recent Advances(Flow-matching)

FLOW MATCHING FOR GENERATIVE MODELING

Yaron Lipman[1,2] **Ricky T. Q. Chen**[1] **Heli Ben-Hamu**[2] **Maximilian Nickel**[1] **Matt Le**[1]
[1]Meta AI (FAIR) [2]Weizmann Institute of Science

ABSTRACT

We introduce a new paradigm for generative modeling built on Continuous Normalizing Flows (CNFs), allowing us to train CNFs at unprecedented scale. Specifically, we present the notion of Flow Matching (FM), a simulation-free approach for training CNFs based on regressing vector fields of fixed conditional probability paths. Flow Matching is compatible with a general family of Gaussian probability paths for transforming between noise and data samples—which subsumes existing diffusion paths as specific instances. Interestingly, we find that employing FM with diffusion paths results in a more robust and stable alternative for training diffusion models. Furthermore, Flow Matching opens the door to training CNFs with other, non-diffusion probability paths. An instance of particular interest is using Optimal Transport (OT) displacement interpolation to define the conditional probability paths. These paths are more efficient than diffusion paths, provide faster training and sampling, and result in better generalization. Training CNFs using Flow Matching on ImageNet leads to consistently better performance than alternative diffusion-based methods in terms of both likelihood and sample quality, and allows fast and reliable sample generation using off-the-shelf numerical ODE solvers.

1 INTRODUCTION

Deep generative models are a class of deep learning algorithms aimed at estimating and sampling from an unknown data distribution. The recent influx of amazing advances in generative modeling, *e.g.*, for image generation Ramesh et al. (2022); Rombach et al. (2022), is mostly facilitated by the scalable and relatively stable training of diffusion-based models Ho et al. (2020); Song et al. (2020b). However, the restriction to simple diffusion processes leads to a rather confined space of sampling probability paths, resulting in very long training times and the need to adopt specialized methods (*e.g.*, Song et al. (2020a); Zhang & Chen (2022)) for efficient sampling.

In this work we consider the general and deterministic framework of Continuous Normalizing Flows (CNFs; Chen et al. (2018)). CNFs are capable of modeling arbitrary probability path and are in particular known to encompass the probability paths modeled by diffusion processes (Song et al., 2021). However, aside from diffusion that can be trained efficiently via, *e.g.*, denoising score matching (Vincent, 2011), no scalable CNF training algorithms are known. Indeed, maximum likelihood training (*e.g.*, Grathwohl et al. (2018)) require expensive numerical ODE simulations, while existing simulation-free methods either involve intractable integrals (Rozen et al., 2021) or biased gradients (Ben-Hamu et al., 2022).

Figure 1: Unconditional ImageNet-128 samples of a CNF trained using Flow Matching with Optimal Transport probability paths.

The goal of this work is to propose Flow Matching (FM), an efficient simulation-free approach to training CNF models, allowing the adoption of general probability paths to supervise CNF training. Importantly, FM breaks the barriers for scalable CNF training beyond diffusion, and sidesteps the need to reason about diffusion processes to directly work with probability paths.

1. Continuous Normalizing Flows 훈련 문제 해결
- 노이즈를 실제 데이터로 점진적으로 변환하는 연속적인 과정을 학습하는 훈련 방식은 변환 과정을 시뮬레이션 해야해서 매우 느리고 비효율적이었다. 따라서 Flow matching이라는 새로운 훈련 방식을 제안했다.

2. Flow-matching 제안
- 변환 과정을 시뮬레이션할 필요 없이, 곧바로 훈련할 수 있게 해주는 훈련 방식을 제안했다. 각 데이터가 다음 순간 어디로 움직여야 하는지를 알려주는 지시 벡터들의 모음인 벡터 필드를 직접 학습하도록 설계되었다.

3. 확률 경로 활용가능
- 우리가 원하는 대로 다양하게 설계 가능해졌다.

IV. Recent Advances(Stable-diffusion 3)

Scaling Rectified Flow Transformers for High-Resolution Image Synthesis

Patrick Esser * Sumith Kulal Andreas Blattmann Rahim Entezari Jonas Müller Harry Saini Yam Levi
Dominik Lorenz Axel Sauer Frederic Boesel Dustin Podell Tim Dockhorn Zion English
Robin Rombach *
Stability AI

Figure 1. High-resolution samples from our 8B rectified flow model, showcasing its capabilities in typography, precise prompt following and spatial reasoning, attention to fine details, and high image quality across a wide variety of styles.

Abstract

Diffusion models create data from noise by inverting the forward paths of data towards noise and have emerged as a powerful generative modeling technique for high-dimensional, perceptual data such as images and videos. Rectified flow is a recent generative model formulation that connects data and noise in a straight line. Despite its better theoretical properties and conceptual simplicity, it is not yet decisively established as standard practice. In this work, we improve existing noise sampling techniques for training rectified flow models by biasing them towards perceptually relevant scales. Through a large-scale study, we demonstrate the superior performance of this approach compared to established diffusion formulations for high-resolution text-to-image synthesis. Additionally, we present a novel transformer-based architecture for text-to-image generation that uses separate weights for the two modalities and enables a bidirectional flow of information between image and text tokens, improving text comprehension, typography, and human preference ratings. We demonstrate that this architecture follows predictable scaling trends and correlates lower validation loss to improved text-to-image synthesis as measured by various metrics and human evaluations. Our largest models outperform state-of-the-art models. Stability AI is considering making experimental data, code, and model weights publicly available.

*Equal contribution . <first.last>@stability.ai.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

1. 기존 Diffusion models의 느린 샘플링 문제 해결
- Rectified Flow 모델의 노이즈 샘플링 기법을 개선하여, 특히 중간 타임스텝에 가중치를 부여함으로써 기존 확산 모델보다 뛰어난 성능과 효율성 달성
2. MM-DiT (Multimodal diffusion Transformer)
- 텍스트와 이미지 토큰 간의 양방향 정보 흐름을 가능하게 하는 새로운 Transformer 기반 아키텍처를 제안하여 텍스트 이해도, 전반적인 이미지 품질을 향상시켰다.
3. 확장 가능성 입증
- 80억 개 매개변수에 이르는 규모로 확장 가능함을 입증

IV. Recent Advances(LRM)

LRM: Large Reconstruction Model for Single Image to 3D

Yicong Hong[1,2*] **Kai Zhang**[1] **Jiuxiang Gu**[1] **Sai Bi**[1] **Yang Zhou**[1]
Difan Liu[1] **Feng Liu**[1] **Kalyan Sunkavalli**[1] **Trung Bui**[1] **Hao Tan**[1]
[1]Adobe Research [2]Australian National Univeristy
mr.yiconghong@gmail.com
{kaiz,jigu,sbi,yazhou,diliu,fengl,sunkaval,bui,hatan}@adobe.com

Abstract

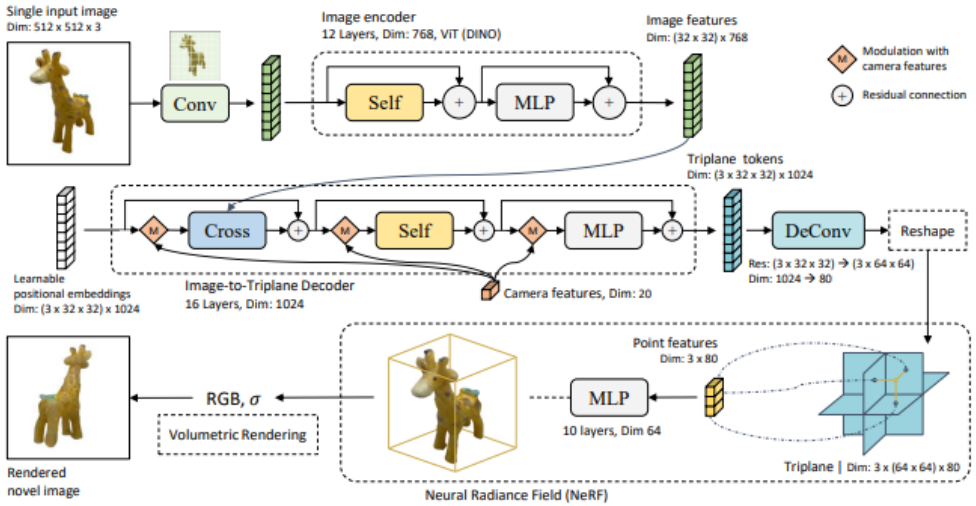
We propose the first Large Reconstruction Model (LRM) that predicts the 3D model of an object from a single input image within just 5 seconds. In contrast to many previous methods that are trained on small-scale datasets such as ShapeNet in a category-specific fashion, LRM adopts a highly scalable transformer-based architecture with 500 million learnable parameters to directly predict a neural radiance field (NeRF) from the input image. We train our model in an end-to-end manner on massive multi-view data containing around 1 million objects, including both synthetic renderings from Objaverse and real captures from MVImgNet. This combination of a high-capacity model and large-scale training data empowers our model to be highly generalizable and produce high-quality 3D reconstructions from various testing inputs, including real-world in-the-wild captures and images created by generative models. Video demos and interactable 3D meshes can be found on our LRM project webpage: https://yiconghong.me/LRM.

1 Introduction

Imagine if we could instantly create a 3D shape from a single image of an arbitrary object. Broad applications in industrial design, animation, gaming, and AR/VR have strongly motivated relevant research in seeking a generic and efficient approach towards this long-standing goal. Due to the underlying ambiguity of 3D geometry in a single view, early learning-based methods usually perform well on specific categories, utilizing the category data prior to infer the overall shape (Yu et al., 2021). Recently, advances in image generation, such as DALL-E (Ramesh et al., 2021) and Stable Diffusion (Rombach et al., 2022), have inspired research that leverages the remarkable generalization capability of 2D diffusion models to enable multi-view supervision (Liu et al., 2023b; Tang et al., 2023). However, many of these methods require delicate parameter tuning and regularization, and their results are limited by the pre-trained 2D generative models. Meanwhile, there are many approaches that rely on per-shape optimization (*e.g.* optimize a NeRF (Mildenhall et al., 2021; Chan et al., 2022; Chen et al., 2022a; Müller et al., 2022; Sun et al., 2022)) to construct a consistent geometry; this process is often slow and impractical.

On the other hand, the great success in natural language processing (Devlin et al., 2018; Brown et al., 2020; Chowdhery et al., 2022) and image processing (Caron et al., 2021; Radford et al., 2021; Alayrac et al., 2022; Ramesh et al., 2022) can be largely credited to three critical factors: (1) using highly scalable and effective neural networks, such as the Transformers (Vaswani et al., 2017), for modeling the data distribution, (2) enormous datasets for learning generic priors, as well as (3) self-supervised-like training objectives that encourage the model to discover the underlying data structure while maintaining high scalability. For instance, the GPT (generative pre-trained transformer) series (Radford et al., 2019; Brown et al., 2020; OpenAI, 2023) build large language models with huge transformer networks, large-scale data, and the simple next-word prediction task. In light of this, we pose the same question for 3D: given sufficient 3D data and a large-scale training framework, ***is it possible to learn a generic 3D prior for reconstructing an object from a single image?***

*Intern at Adobe Research.



1. 최초의 대규모 3D 재규성 모델

- Transformer 기반의 확장 가능한 아키텍처를 사용하여 단일 이미지로부터 NeRF를 직접 예측하는 5억 개 이상의 파라미터를 가진 첫 번째 대규모 3D 재구성 모델이다.

2. 대규모 멀티뷰 데이터 기반 학습

- 약 1백만 개의 객체로 구성된 방대한 멀티뷰 데이터셋에서 end-to-end 방식으로 학습되었다.

IV. Recent Advances(LGM)

LGM: Large Multi-View Gaussian Model for High-Resolution 3D Content Creation

Jiaxiang Tang[1]*, Zhaoxi Chen[2], Xiaokang Chen[1], Tengfei Wang[3], Gang Zeng[1], and Ziwei Liu[2]

[1] National Key Lab of General AI, Peking University
[2] S-Lab, Nanyang Technological University
[3] Shanghai AI Lab

https://me.kiui.moe/lgm

Fig. 1: Our method generates **high-resolution** 3D Gaussians in **5 seconds** from single-view images or texts.

Abstract. 3D content creation has achieved significant progress in terms of both quality and speed. Although current feed-forward models can produce 3D objects in seconds, their resolution is constrained by the intensive computation required during training. In this paper, we introduce **Large Multi-View Gaussian Model (LGM)**, a novel framework designed to generate high-resolution 3D models from text prompts or single-view images. Our key insights are two-fold: **1) 3D Representation:** We propose multi-view Gaussian features as an efficient yet powerful representation, which can then be fused together for differentiable rendering. **2) 3D Backbone:** We present an asymmetric U-Net as a high-throughput backbone operating on multi-view images, which can be

* Work done while visiting S-Lab, Nanyang Technological University.

1. 고해상도 및 고효율 3D 콘텐츠 생성
- 텍스트 프롬프트나 단일 뷰 이미지로부터 고행상도 3D 모델을 단 5초 만에 생성하는 것을 목표로 한다. 기존 feed-forward 방식 3D 생성 모델이 겪었던 낮은 행상도와 느린 처리 속도 문제를 해결하며, 512 해상도에서 3D Gaussians를 훈련하고 생성할 수 있게 한다.

2. 효율적인 3D 표현 및 백본 아키텍처
- 3D Representation으로 multi-view Gaussian features를 사용하여, 기존 NeRF 기반 모델보다 효율적인 표현력과 빠른 렌더링 속도를 확보. 그리고 3D backbone으로 asymmetric U-net을 사용하여 Multi-view 이미지를 처리하고 Gaussians를 예측 및 융합함으로써, 고해상도 학습과 빠른 처리를 가능하게 했습니다.

IV. Recent Advances(CityGaussian)

CityGaussian: Real-time High-quality Large-Scale Scene Rendering with Gaussians

Yang Liu[1,2], He Guan[1,2], Chuanchen Luo[4], Lue Fan[1,2], Naiyan Wang[5], Junran Peng[6] ✉, and Zhaoxiang Zhang[1,2,3] ✉

[1] NLPR, MAIS, Institute of Automation, Chinese Academy of Sciences
[2] University of Chinese Academy of Sciences
[3]Centre for Artificial Intelligence and Robotic [4]Shandong University
[5]TuSimple [6]University of Science and Technology Beijing
{liuyang2022, lue.fan, zhaoxiang.zhang}@ia.ac.cn,
chuanchen.luo@sdu.edu.cn, {sylcito,winsty}@gmail.com, jrpeng4ever@126.com

Abstract. The advancement of real-time 3D scene reconstruction and novel view synthesis has been significantly propelled by 3D Gaussian Splatting (3DGS). However, effectively training large-scale 3DGS and rendering it in real-time across various scales remains challenging. This paper introduces CityGaussian (CityGS), which employs a novel divide-and-conquer training approach and Level-of-Detail (LoD) strategy for efficient large-scale 3DGS training and rendering. Specifically, the global scene prior and adaptive training data selection enables efficient training and seamless fusion. Based on fused Gaussian primitives, we generate different detail levels through compression, and realize fast rendering across various scales through the proposed block-wise detail levels selection and aggregation strategy. Extensive experimental results on large-scale scenes demonstrate that our approach attains state-of-the-art rendering quality, enabling consistent real-time rendering of large-scale scenes across vastly different scales. Our project page is available at https://dekuliutesla.github.io/citygs/.

Keywords: Large-Scale Scene Reconstruction · Novel View Synthesis · 3D Gaussian Splatting

1 Introduction

3D large-scale scene reconstruction, as a pivotal component in AR/VR [6, 11], aerial surveying [36], smart city [4, 8], and autonomous driving [34], has drawn extensive attention from academia and industry in recent decades. Such a task pursues high-fidelity reconstruction and real-time rendering at different scales for large areas that typically span over 1.5 km^2 [36]. In the past few years, this field has been dominated by neural radiance fields (NeRF) [21] based methods. Representative works include Block-NeRF [34], BungeeNeRF [41], and ScaNeRF [40]. But they still lack fidelity in details or exhibit sluggish performance.

Recently, 3D Gaussian Splatting (3DGS) [12] emerged as a promising alternative solution. In contrast to NeRF, it employs explicit 3D Gaussians as

1. Divde-and-Conquer 방식 학습
- 대규모 장면의 3D Gaussian Splatting 학습을 위해 전체 장면을 공간적으로 인접한 블록으로 나누고 병렬적으로 학습. 기존 3DGS가 대규모 장면에서 겪던 메모리 과부하 문제를 해결.
2. Level-of-Detail 렌더링 전략
- 3D Gaussians를 렌더링할 때 발생하는 게산 부담을 줄이기 위해 제안. 장면을 블록 단위로 나누고, 카메라의 거리에 따라 각 블록에 적합한 Detail Level의 Gaussians를 동적으로 선택하여 렌더링에 사용.

IV. Recent Advances(DiffSplat)

DIFFSPLAT: REPURPOSING IMAGE DIFFUSION MODELS FOR SCALABLE GAUSSIAN SPLAT GENERATION

Chenguo Lin[1], **Panwang Pan**[†2], **Bangbang Yang**[2], **Zeming Li**[2], **Yadong Mu**[‡1]
[1]Peking University, [2]ByteDance
https://chenguolin.github.io/projects/DiffSplat

ABSTRACT

Recent advancements in 3D content generation from text or a single image struggle with limited high-quality 3D datasets and inconsistency from 2D multi-view generation. We introduce DIFFSPLAT, a novel 3D generative framework that natively generates 3D Gaussian splats by taming large-scale text-to-image diffusion models. It differs from previous 3D generative models by effectively utilizing web-scale 2D priors while maintaining 3D consistency in a unified model. To bootstrap the training, a lightweight reconstruction model is proposed to instantly produce multi-view Gaussian splat grids for scalable dataset curation. In conjunction with the regular diffusion loss on these grids, a 3D rendering loss is introduced to facilitate 3D coherence across arbitrary views. The compatibility with image diffusion models enables seamless adaptions of numerous techniques for image generation to the 3D realm. Extensive experiments reveal the superiority of DIFFSPLAT in text- and image-conditioned generation tasks and downstream applications. Thorough ablation studies validate the efficacy of each critical design choice and provide insights into the underlying mechanism.

1 INTRODUCTION

Generating 3D content from a single image or text is a long-standing challenge with a wide range of applications, such as game design, digital arts, human avatars, and virtual reality. It is a highly ill-posed problem that requires reasoning the unseen parts of any object in the 3D space only from a single view or textual descriptions, posing a great challenge to both fidelity and generalizability.

With the development of diffusion generative models (Sohl-Dickstein et al., 2015; Ho et al., 2020), recent works train conditional 3D generative networks directly on datasets of various 3D representations (Nichol et al., 2022; Jun & Nichol, 2023; Cao et al., 2024; He et al., 2024; Zhang et al., 2024b), as demonstrated in Figure 1 (1), or only using 2D supervision with the help of differentiable rendering techniques (Anciukevičius et al., 2023; Karnewar et al., 2023b; Szymanowicz et al., 2023; Xu et al., 2024d) as in Figure 1 (2). Despite 3D consistency, they are limited by the supervision scale and can't utilize 2D priors from abundant pre-trained models. Current advanced generalizable 3D content creation methods (Li et al., 2024a; Wang et al., 2024; Tang et al., 2024) reconstruct implicit 3D representations from generated multi-view images using pretrained 2D diffusion models (Wang & Shi, 2023; Shi et al., 2023; Voleti et al., 2024), as illustrated in Figure 1 (3). Although these two-stage methods can reconstruct high-quality 3D content from multi-view posed images, the synthesis pipeline collapses and fails to produce faithful results when generated images from upstreamed 2D multi-view diffusion models are of poor quality or inconsistency.

To overcome the drawbacks of previous works, we present DIFFSPLAT, a novel 3D generative framework that exhibits multi-view consistency and effectively leverages generative priors from large-scale image datasets. We adopt 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) as the 3D content representation for its efficient rendering and quality balance. Instead of relying on time-consuming per-instance optimization to obtain 3D datasets for training (He et al., 2024; Zhang et al., 2024b), we represent a 3D object by a set of well-structured splat 2D grids. During the training stage, these grids can be instantly regressed from multi-view images in less than 0.1 seconds, facilitating scalable and high-quality 3D dataset curation. Each Gaussian splat in 2D grids holds properties that imply object texture and structure. Noting that image diffusion models trained on web-scale datasets are capable

†: Project lead; ‡: Corresponding author.

1. 2D diffusion Models의 재활용
- 기존 대규모 Text-to-Image Diffusion Models를 활용하여 3D Gaussian Splat을 직접 생성하는 새로운 3D 생성 프레임워크를 제시. 방대한 2D 사전 지식을 효과적으로 활용하고 3D 일관성을 유지.

2. 학습 목적 함수 강화
- 3D Rendering Loss를 도입하여 3D 컨텐츠의 렌더링 품질을 보장한다. 일반적인 Diffusion Loss와 같이 사용한다. 따라서, 2D diffusion models의 특성을 활용하면서도 3D 공간에서의 일관되고 사실적인 객체 생성을 가능하게 한다.