

Radiative Gaussian Splatting for Efficient X-ray Novel View Synthesis

Yuanhao Cai¹, Yixun Liang², Jiahao Wang¹, Angtian Wang¹,
Yulun Zhang^{3,†}, Xiaokang Yang³, Zongwei Zhou^{1,†}, and Alan Yuille¹

¹ Johns Hopkins University, ² HKUST(GZ), ³ Shanghai Jiao Tong University

Abstract. X-ray is widely applied for transmission imaging due to its stronger penetration than natural light. When rendering novel view X-ray projections, existing methods mainly based on NeRF suffer from long training time and slow inference speed. In this paper, we propose a 3D Gaussian splatting-based framework, namely X-Gaussian, for X-ray novel view synthesis. Firstly, we redesign a radiative Gaussian point cloud model inspired by the isotropic nature of X-ray imaging. Our model excludes the influence of view direction when learning to predict the radiation intensity of 3D points. Based on this model, we develop a Differentiable Radiative Rasterization (DRR) with CUDA implementation. Secondly, we customize an Angle-pose Cuboid Uniform Initialization (ACUI) strategy that directly uses the parameters of the X-ray scanner to compute the camera information and then uniformly samples point positions within a cuboid enclosing the scanned object. Experiments show that our X-Gaussian outperforms state-of-the-art methods by **6.5 dB** while enjoying less than **15%** training time and over **73×** inference speed. The application on sparse-view CT reconstruction also reveals the practical values of our method. <https://github.com/caiyuanhao1998/X-Gaussian>

1 Introduction

X-ray novel view synthesis (NVS) aims to create X-ray projections of an object from new viewpoints that are not originally captured, using only existing projections scanned from different view directions. As we know, X-ray has stronger penetrating power to capture internal structures of imaged objects and is thus widely applied in medical imaging [13, 14, 16, 19, 20, 43]. Yet, X-ray is harmful to human body due to its powerful ionizing radiation, especially when the dose of X-ray increases. Improving NVS techniques can help reduce the exposure to X-rays and provide comprehensive viewpoints of imaged parts for doctors and downstream tasks such as CT reconstruction. Thus, X-ray NVS is very important and valuable. We study this task in the circular cone beam X-ray scanning scenario [7, 8, 11, 35, 39, 44, 56, 59].

† = corresponding authors.

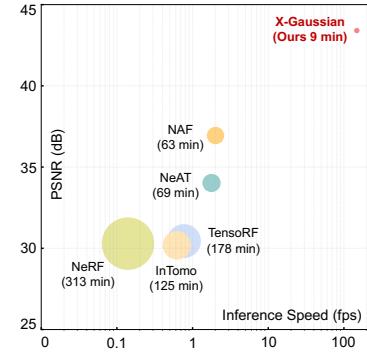


Fig. 1: PSNR-minute-fps comparison. The radius of circle represents the training time (minutes). Our method is the most efficient.

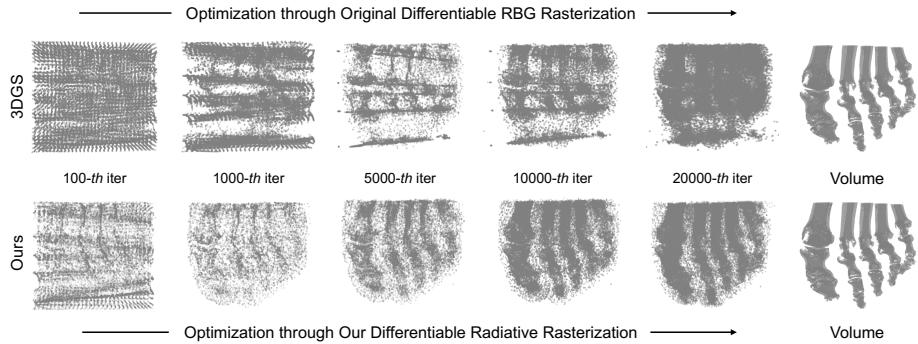


Fig. 2: Point cloud visualization of the original 3DGS [26] (top) and our X-Gaussian (bottom). We visualize the positions and opacities of the Gaussian point clouds at different training iterations. We also visualize the volume of foot as a reference. Note that the volume is not the ground truth of point clouds. Our X-Gaussian can better represent the detailed structures than 3DGS, showing faster and better convergence.

Existing methods are mainly based on neural radiance fields (NeRF) [37]. They usually employ a multi-layer perceptron (MLP) to learn the mapping from the point position to the radiodensity and then create projections by volume rendering along rays. This ray tracing scheme is time-consuming because it needs to sample many 3D points and then compute them for every single ray, slowing down the training and inference processes. Even the recent most efficient NeRF-based method [60] still requires over an hour for training and yields suboptimal results at a slow inference speed of 2 fps, as shown in Fig. 1. This increases the waiting time of patients and doctors, leading to low diagnostic efficiency.

Recently, 3D Gaussian splatting (3DGS) [26] has demonstrated promising reconstruction quality while enjoying much faster inference speed than NeRF-based algorithms in RGB domain, which motivates us to follow this technical route. However, due to the fundamental differences between X-ray and natural light imaging, directly applying the original 3DGS to X-ray NVS may encounter two issues. **Firstly**, the spherical harmonics (SH) in RGB 3DGS is not suitable for modeling the X-ray radiation intensity of 3D points. Specifically, natural light imaging relies on the reflection off the surface. The color of a 3D point is anisotropic and view-dependent. Based on this nature, the original Gaussian point cloud model uses SH to fit the illumination distribution. In contrast, X-rays penetrate the object and attenuate, thereby forming an image. Given specific X-rays, the radiation intensity of a 3D point depends on its radiodensity and is independent to the view direction, which means the point radiation intensity is isotropic. **Secondly**, the original point cloud initialization algorithm, structure-from-motion (SfM) [46], is also not suitable for X-ray imaging. Compared to RGB images, X-ray images are grayscale and their contrast is lower. Additionally, different layers of an object may overlap on the same position of the projection due to the transmission imaging nature of X-rays. These two problems degrade the accuracy of feature detection and matching in SfM. Meanwhile, running SfM is time-consuming, which prolongs the training process of Gaussian point clouds.

To address the above issues, we propose a novel 3DGS-based method, X-Gaussian, for X-ray NVS. Our X-Gaussian composes two key techniques. **Firstly**, we redesign a radiative Gaussian point cloud model inspired by the isotropic property of X-ray imaging. We present a Radiation Intensity Response Function (RIRF) to replace the SH function of the original 3DGS. Different from SH, our RIRF excludes the influence of view direction. To this end, it adopts the inner product between a learnable vector representing the inherent point features and a set of basis weights to fit the radiation intensity of a 3D point. Based on this point cloud model, we further develop a Differentiable Radiative Rasterization (DRR) with a CUDA implementation to render novel projections. **Secondly**, we customize an Angle-pose Cuboid Uniform Initialization (ACUI) strategy for camera calibration parameters and Gaussian point clouds. Our ACUI first exploits the parameters of the X-ray scanner to compute the intrinsic and extrinsic matrices. Then we set up a cuboid that can completely enclose the scanned object. Within this cuboid, we uniformly sample 3D points at intervals to initialize the center positions of the Gaussian point clouds. Free from running the SfM algorithm, our ACUI significantly reduces the training time. Equipped with the two proposed techniques, our X-Gaussian enjoys faster convergence, better performance, and shorter running time than state-of-the-art (SOTA) algorithms, as shown in Figs. 1 and 2. Surprisingly, X-Gaussian outperforms SOTA methods by **6.5 dB** while enjoying **73 \times** inference speed and **7 \times** training speed.

The main contributions of this work can be summarized as follows:

- We propose a novel 3D Gaussian splatting-based framework, X-Gaussian, for X-ray novel view synthesis. To our knowledge, this is the first attempt to explore the potential of Gaussian splatting in X-ray neural rendering.
- We design a radiative Gaussian point cloud model with a differentiable radiative rasterization based on the isotropic nature of X-ray imaging.
- We present an angle-pose cuboid uniform initialization strategy for Gaussian point clouds and camera calibration in circular cone beam X-ray scanning.
- Our X-Gaussian significantly outperforms SOTA NeRF-based methods with much faster speed. Experiments also show that our method can improve the performance of sparse-view CT reconstruction, showing its practical values.

2 Related Work

2.1 Neural Radiance Field

NeRF [37] learns an implicit neural scene representation of color and volume density, given the position of a 3D point and view direction. It has achieved great success in NVS and inspired an explosion of follow-up papers to improve its quality [3–5, 23, 48] and speed [10, 12, 22, 31, 38, 41, 54]. For example, Instant-NGP [38] adopts hash tables as the encoder to allow small MLP for fast training and inference. Some later works extend the application domain of NeRF from natural light to X-rays [9, 15, 58, 60]. For instance, NAF [60] follows the settings of Instant-NGP to learn the implicit mapping from 3D position to attenuation. Yet, the ray tracing and volume rendering schemes are time-consuming, which limits the training and inference speed of NeRF-based X-ray NVS algorithms.

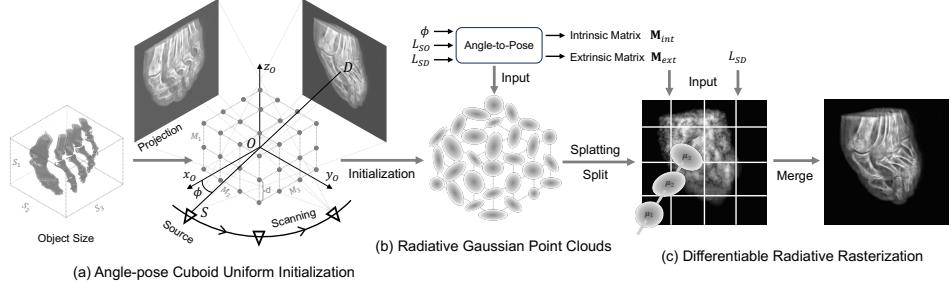


Fig. 3: Pipeline of our method. (a) Angle-pose Cuboid Uniform Initialization (ACUI) strategy uses the parameters of X-ray scanner to compute intrinsic and extrinsic matrices, and samples center points for 3D Gaussians. (b) Our radiative Gaussian point cloud model learns to predict the radiation intensity of 3D points. (c) Based on our Gaussian model, we develop a GPU-friendly Differentiable Radiative Rasterization (DRR).

2.2 Gaussian Splatting

3DGS [26] represents scenes using millions of 3D Gaussian point clouds. This approach is fundamentally different from NeRF-based algorithms by employing an explicit representation coupled with highly parallelized rasterization workflows. These features enable more efficient computation and rendering processes. Hence, 3DGS has achieved great success in several fields, including 3D Generation [21, 29, 32, 47, 55], Dynamic Scene Modeling [34, 50, 53], SLAM [25, 36, 52, 57], Inverse Rendering [24, 33, 51], *etc.* However, most applications of 3DGS are focused on natural scenes with RGB colors. The potential of 3DGS in X-ray imaging still remains under-explored. Our goal is to fill this research gap.

3 Method

The pipeline of our X-Gaussian is shown in Fig. 3. Firstly, we design an Angle-pose Cuboid Uniform Initialization (ACUI) to compute the intrinsic and extrinsic matrices from the parameters of X-ray scanner, as illustrated in Fig. 3 (a). Then ACUI uniformly samples 3D points within a cuboid that can completely enclose the scanned object to initialize the center positions of our radiative Gaussian point clouds in Fig. 3 (b). Given a view direction, the 3D point clouds undergo our Differentiable Radiative Rasterization (DRR) to derive the rendered image, as depicted in Fig. 3 (c). In this section, we will introduce our radiative Gaussian point cloud model and DRR processing first and then the ACUI strategy.

3.1 Radiative Gaussian Point Cloud Model

An object can be represented by a set of basic Gaussian point clouds \mathcal{G} as

$$\mathcal{G} = \{G_i(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i) \mid i = 1, 2, \dots, N_p\}, \quad (1)$$

where G_i refers to the i -th Gaussian point cloud. Its center position, covariance, and opacity are defined as $\boldsymbol{\mu}_i \in \mathbb{R}^3$, $\boldsymbol{\Sigma}_i \in \mathbb{R}^{3 \times 3}$, and $\alpha_i \in \mathbb{R}$. $\boldsymbol{\Sigma}_i$ is represented

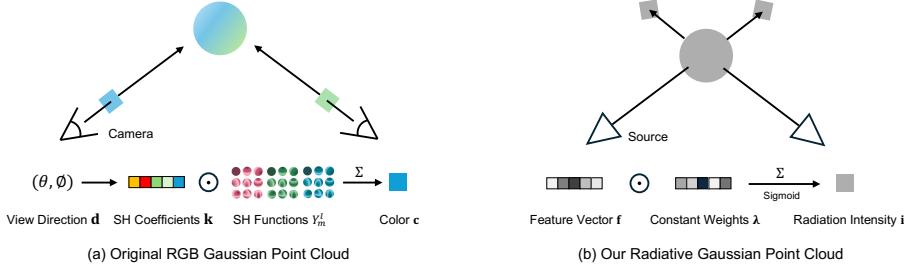


Fig. 4: Comparison between the Gaussian point cloud models of the original 3DGS and our X-Gaussian. (a) The original RGB Gaussian point cloud model uses spherical harmonics (SH) to simulate the anisotropic natural light distribution and view-dependent color. (b) Our radiative Gaussian point cloud model employs the weighted sum of point features to fit the isotropic X-ray penetration and view-independent radiation intensity.

by a rotation matrix $\mathbf{R}_i \in \mathbb{R}^3$ and a scaling matrix $\mathbf{S}_i \in \mathbb{R}^3$ as $\Sigma_i = \mathbf{R}_i \mathbf{S}_i \mathbf{S}_i^\top \mathbf{R}_i^\top$. μ_i , Σ_i , α_i , \mathbf{R}_i , and \mathbf{S}_i are learnable parameters. Besides these basic attributes, each Gaussian point cloud also employs additional learnable parameters to fit different imaging scenarios, *e.g.*, natural light imaging and X-ray imaging.

We first review the original RGB Gaussian point cloud model [26] in natural light imaging. As shown in Fig. 4 (a), the color of a 3D point is represented by spherical harmonics (SH). The point color is anisotropic and changes with the view direction. Each Gaussian point cloud learns to predict the SH coefficients $\mathbf{k} = \{k_l^m | 0 \leq l \leq L, -l \leq m \leq l\} \in \mathbb{R}^{(L+1)^2 \times 3}$, where each $k_l^m \in \mathbb{R}^3$ is a set of 3 coefficients corresponding to the RGB components. L is the degree of SH. Then the point color $\mathbf{c} \in \mathbb{R}^3$ at the view direction $\mathbf{d} = (\theta, \phi)$ is derived by

$$\mathbf{c}(\mathbf{d}, \mathbf{k}) = \sum_{l=0}^L \sum_{m=-l}^l k_l^m Y_l^m(\theta, \phi), \quad (2)$$

where $Y_l^m : \mathbb{S}^2 \rightarrow \mathbb{R}$ is the SH function that maps points on the sphere to real numbers. Please refer to the supplementary for its detailed formulation.

Although 3DGS [26] achieves fast inference speed and good performance in natural light imaging, the RGB Gaussian point cloud model is not suitable for X-ray scenarios due to the fundamental differences between natural light imaging and X-ray imaging. Natural light imaging relies on the reflection off the surface of object. The anisotropic color modeled by SH is view-dependent. *e.g.*, in Fig. 4 (a), the point color is blue from the left viewpoint and green from the right viewpoint. In contrast, X-ray imaging is based on the attenuation when penetrating the object. The degree of attenuation depends on the isotropic radiodensity property. Thus, the radiation intensity of a 3D point is view-independent.

In light of the above analysis, we redesign our radiative Gaussian point cloud model. Different from the original 3DGS that uses SH to fit the color information for each point, our model introduces a Radiation Intensity Response Function (RIRF) to predict the radiation intensity of the 3D point. As illustrated in Fig. 4 (b), each Gaussian point cloud learns a feature vector $\mathbf{f} \in \mathbb{R}^{N_f}$ to represent its inherent radiative properties. Subsequently, the radiation intensity $\mathbf{i} \in \mathbb{R}$ of the

center point of a 3D Gaussian at any view direction is modeled by RIRF as

$$\mathbf{i}(\mathbf{f}) = \text{RIRF}(\mathbf{f}) = \text{Sigmoid}(\boldsymbol{\lambda} \odot \mathbf{f}), \quad (3)$$

where the Sigmoid function activates and normalizes the radiation intensity. $\boldsymbol{\lambda} \in \mathbb{R}^{N_f}$ is a set of constant weights controlling the importance of each component of \mathbf{f} . Then the set of our radiative Gaussian point clouds \mathcal{G}_x is formulated as

$$\mathcal{G}_x = \{G_i(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, \mathbf{f}_i) \mid i = 1, 2, \dots, N_p\}, \quad (4)$$

where $\mathbf{f}_i \in \mathbb{R}^{N_f}$ denotes the feature vector of the i -th Gaussian point cloud. Please note that Eq. (3) excludes the influence of the view direction $\mathbf{d} = (\theta, \phi)$, which matches the isotropic nature of X-ray imaging. Meanwhile, Eq. (3) is free from the complex computation of SH function. Hence, the forward and backward processes of our X-Gaussian are much faster than those of the original 3DGs.

3.2 Differentiable Radiative Rasterization

Based on our radiative Gaussian point cloud, we develop a Differentiable Radiative Rasterization (DRR), as shown in Fig. 3 (c). We first summarize the overall DRR processing F_{DRR} and then describe its details. DRR is represented as

$$\mathbf{I} = F_{\text{DRR}}(\mathbf{M}_{\text{ext}}, \mathbf{M}_{\text{int}}, \{G_i(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, \mathbf{f}_i) \mid i = 1, 2, \dots, N_p\}), \quad (5)$$

where $\mathbf{I} \in \mathbb{R}^{H \times W}$ denotes the rendered image, $\mathbf{M}_{\text{ext}} \in \mathbb{R}^{4 \times 4}$ represents the extrinsic matrix, and $\mathbf{M}_{\text{int}} \in \mathbb{R}^{4 \times 3}$ refers to the intrinsic matrix. Subsequently, we introduce the details of F_{DRR} . To begin with, the possibility value of the i -th Gaussian distribution at the 3D point position $\mathbf{x} \in \mathbb{R}^3$ is formulated as

$$P(\mathbf{x} | \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) = \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_i)^\top \boldsymbol{\Sigma}_i^{-1}(\mathbf{x} - \boldsymbol{\mu}_i)\right). \quad (6)$$

Then we project the 3D Gaussians to the 2D detector plane for subsequent rendering. $\boldsymbol{\mu}_i$ is firstly transferred from the world coordinate system to the camera coordinate system and then projected to the image coordinate system as

$$\tilde{\mathbf{t}}_i = \begin{bmatrix} \mathbf{t}_i \\ 1 \end{bmatrix} = \mathbf{M}_{\text{ext}} \tilde{\boldsymbol{\mu}}_i = \mathbf{M}_{\text{ext}} \begin{bmatrix} \boldsymbol{\mu}_i \\ 1 \end{bmatrix}, \quad \tilde{\mathbf{u}}_i = \begin{bmatrix} \mathbf{u}_i \\ 1 \end{bmatrix} = \mathbf{M}_{\text{int}} \tilde{\mathbf{t}}_i = \mathbf{M}_{\text{int}} \begin{bmatrix} \mathbf{t}_i \\ 1 \end{bmatrix}, \quad (7)$$

where $\mathbf{t}_i = (t_x, t_y, t_z) \in \mathbb{R}^3$ is the camera coordinate of $\boldsymbol{\mu}_i$ and $\mathbf{u}_i \in \mathbb{R}^2$ is the image coordinate of $\boldsymbol{\mu}_i$. $\tilde{\mathbf{u}}_i$, $\tilde{\mathbf{t}}_i$, and $\tilde{\boldsymbol{\mu}}_i$ are the homogeneous coordinates of \mathbf{u}_i , \mathbf{t}_i , and $\boldsymbol{\mu}_i$, respectively. Subsequently, we transfer the 3D covariance matrix $\boldsymbol{\Sigma}_i$ to its counterpart $\boldsymbol{\Sigma}'_i \in \mathbb{R}^{3 \times 3}$ in the camera coordinate system as

$$\boldsymbol{\Sigma}'_i = \mathbf{J}_i \mathbf{W}_i \boldsymbol{\Sigma}_i \mathbf{W}_i^\top \mathbf{J}_i^\top, \quad (8)$$

where $\mathbf{J}_i \in \mathbb{R}^{3 \times 3}$ is the Jacobian of the affine approximation of the projective transformation. $\mathbf{W}_i \in \mathbb{R}^{3 \times 3}$ is the viewing transformation. We derive them by

$$\mathbf{J}_i = \begin{bmatrix} \frac{L_{SD}}{t_z} & 0 & -\frac{L_{SD} t_x}{t_z^2} \\ 0 & \frac{L_{SD}}{t_z} & -\frac{L_{SD} t_y}{t_z^2} \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{W}_i = \begin{bmatrix} -\sin \phi & \cos \phi & 0 \\ 0 & 0 & -1 \\ -\cos \phi & -\sin \phi & 0 \end{bmatrix}, \quad (9)$$

where L_{SD} represents the distance between the X-ray source and detector. ϕ refers to the azimuth angle of the source. Following [26, 30, 61], we obtain the 2D

covariance matrix $\Sigma_i'' \in \mathbb{R}^{2 \times 2}$ by skipping the third row and column of Σ_i' . Then the 2D projection is partitioned into non-overlapping tiles. The 3D Gaussians (μ_i, Σ_i) are assigned to different tiles according to their 2D projections (u_i, Σ_i'') , as shown in the left image of Fig. 3 (c). These 3D Gaussians are sorted by the distances to the 2D detector. Then the intensity $I(p) \in \mathbb{R}$ at pixel p is obtained by blending N ordered points overlapping the pixel in the corresponding tile as

$$I(p) = \sum_{j \in \mathcal{N}} i_j \sigma_j \prod_{k=1}^{j-1} (1 - \sigma_k), \quad \sigma_j = \alpha_j P(\mathbf{x}_j | \mu_j, \Sigma_j), \quad (10)$$

where \mathbf{x}_j is the j -th intersection 3D point of the X-ray landing on pixel p and the Gaussian point clouds in 3D space. i_j is the radiation intensity of \mathbf{x}_j .

Optimization. Eventually, the training objective \mathcal{L} is the weighted sum of \mathcal{L}_1 loss and SSIM loss between the rendered and ground-truth projection images as

$$\mathcal{L} = (1 - \gamma) \mathcal{L}_1 + \gamma \mathcal{L}_{\text{SSIM}}, \quad (11)$$

where γ is a hyperparameter balancing the importances of the two loss terms. By minimizing Eq. (11), we can optimize the attributes of 3D Gaussians, *i.e.*, μ_i , Σ_i , α_i , and \mathbf{f}_i in Eq. (5). N_p is adjusted by the adaptive control [26]. The optimization process is visualized in Fig. 2 and the video file in supplementary.

Compared to the RGB rasterization in 3DGS [26], our DRR avoids the complex computations related to the view direction in the forward and backward processes, thereby enjoying cheaper training costs and faster inference speed.

3.3 Angle-pose Cuboid Uniform Initialization

At the beginning of training, we need to initialize the parameters in Eq. (5) for rasterization. Specifically, Σ_i , α_i , and \mathbf{f}_i are randomly initialized. In natural light imaging, the original 3DGS [26] adopts the SfM [46] algorithm to compute the initial μ_i , N_p , \mathbf{M}_{ext} , and \mathbf{M}_{int} . SfM detects and matches features from multi-view images. It is not suitable for X-ray imaging due to two reasons. Firstly, X-ray images are grayscale and low-contrast. Secondly, different layers of an object may overlap on the same positions of the projection. These two problems degrade the accuracy of feature detection and matching in SfM. Besides, running the SfM algorithm usually requires a long time, which prolongs the training process.

To address these issues, we customize an Angle-pose Cuboid Uniform Initialization (ACUI) strategy for circular cone beam X-ray scanning scenario where a scanner emits cone-shaped X-ray beams and captures projections at equal angular intervals. As shown in Fig. 3 (a), ACUI uses the parameters of X-ray scanner to compute the extrinsic matrix \mathbf{M}_{ext} and intrinsic matrix \mathbf{M}_{int} as

$$\mathbf{M}_{ext} = \begin{bmatrix} -\sin \phi & \cos \phi & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -\cos \phi & -\sin \phi & 0 & L_{SO} \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{M}_{int} = \begin{bmatrix} L_{SD} & 0 & W/2 & 0 \\ 0 & L_{SD} & H/2 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad (12)$$

where L_{SO} represents the distance between the X-ray source and the scanned object. The elevation angle of the X-ray source is set to zero and remains unchanged. The next step of ACUI is to initialize the center positions of 3D Gaussians. Although the precise shape of the scanned object is not given at the

Method	Infer Speed	Train Time	Chest	Foot	Head	Abdomen	Pancreas	Average
InTomo [58]	0.62 fps	125 min	28.948 0.9915	39.482 0.9979	34.832 0.9977	27.641 0.9646	20.031 0.8537	30.187 0.9611
NeRF [37]	0.14 fps	313 min	36.157 0.9988	41.053 0.9989	29.760 0.9991	24.620 0.9559	19.853 0.8560	30.289 0.9617
TensoRF [10]	0.77 fps	178 min	23.609 0.9402	37.728 0.9929	34.429 0.9879	27.382 0.8730	29.235 0.8031	30.477 0.9194
NeAT [42]	1.78 fps	69 min	40.765 0.9990	38.236 0.9963	27.738 0.9295	26.741 0.8563	37.526 0.9017	34.201 0.9366
NAF [60]	2.01 fps	63 min	42.366 0.9993	38.353 0.9913	30.174 0.9531	37.590 0.9855	36.228 0.8844	36.942 0.9627
X-Gaussian	148 fps	9 min	43.887 0.9998	42.153 0.9997	41.579 0.9997	45.762 0.9999	43.640 0.9976	43.404 0.9993

Table 1: Quantitative results on the novel view synthesis task. The average inference speed and training time of all scenes evaluated on an RTX 8000 GPU are reported. In the cell of the results of each scene, PSNR (upper) and SSIM (lower) are listed.

beginning, the scanning space can be approximated. We set up a cuboid with size $S_1 \times S_2 \times S_3$ (mm) that can completely enclose the object. The center of this cuboid is also the center of the object and the origin of the world coordinate system. We divide this cuboid by a grid with size $M_1 \times M_2 \times M_3$ (voxel). Then we uniformly sample points within the grid at interval $d \in \mathbb{R}$ as

$$\mathcal{P} = \left\{ \left(\frac{n_1 S_1 d}{M_1}, \frac{n_2 S_2 d}{M_2}, \frac{n_3 S_3 d}{M_3} \right) \mid -\lceil \frac{M_i}{2d} \rceil - 1 \leq n_i \leq \lceil \frac{M_i}{2d} \rceil + 1, i = 1, 2, 3 \right\}, \quad (13)$$

where $n_i \in \mathbb{Z}$. Then we use the size and elements of \mathcal{P} to initialize N_p and μ_i . Avoiding running SfM, ACUI allows X-Gaussian to enjoy a faster training speed.

4 Experiments

4.1 Experimental Settings

Dataset. Following NAF [60], we adopt the public datasets of human organ CTs, *i.e.*, LIDC-IDRI [2] and the open scientific visualization dataset [28], to evaluate our method. The test scenes include chest, foot, head, abdomen, and pancreas. We adopt the open-source tomographic toolbox TIGRE [6] to capture 100 projections with 3% noise in the range of $0 \sim 180^\circ$. In the NVS task, 50 projections are used for training and the other 50 projections are used for testing. The CT volumes are used for testing in the sparse-view CT reconstruction task.

Implementation Details. Our X-Gaussian is implemented by PyTorch [40] and CUDA [18]. The model is trained with the Adam optimizer [27] ($\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1 \times 10^{-15}$) for 2×10^4 iterations. The learning rate for point cloud position is initially set to 1.9×10^{-4} and exponentially decays to 1.9×10^{-6} . The learning rates for point feature, opacity, scaling, and rotation are

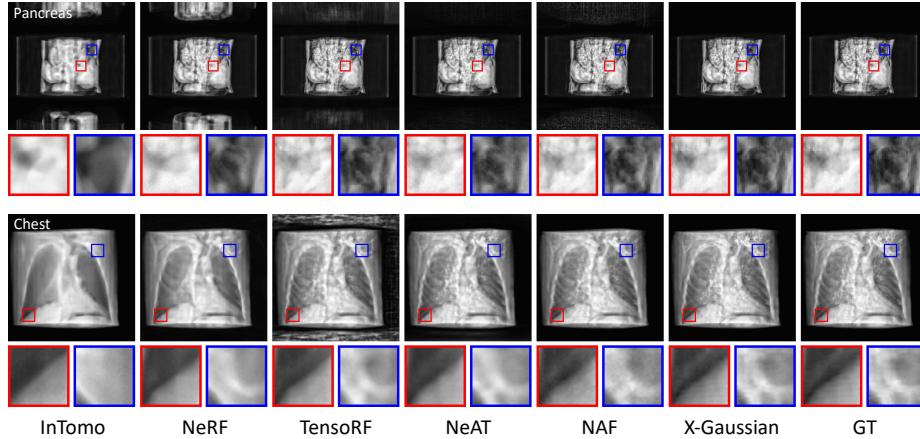


Fig. 5: Qualitative results of novel view synthesis on the scenes of pancreas (top) and chest (bottom). Our X-Gaussian yields clearer results. Please zoom in for a better view.

set to 2×10^{-3} , 8×10^{-3} , 5×10^{-3} , and 1×10^{-3} . γ in Eq. (11) is set to 0.2. We adopt peak signal-to-noise ratio (PSNR) and structural similarity index measure (SSIM) [49] to evaluate the performance. Frames per second (fps) is used to measure the inference speed. Experiments are conducted on an RTX 8000 GPU.

4.2 Novel View Synthesis

Quantitative Results. Tab. 1 shows the quantitative comparisons between our X-Gaussian and five SOTA NeRF-based algorithms, including InTomo [58], NeRF [37], TensorRF [10], NeAT [42], and NAF [60] on the NVS task.

We report the average inference speed and training time of different methods on all scenes. In the cell of the results of each scene, PSNR (upper entry in the cell) and SSIM (lower entry in the cell) are listed. As can be observed that our X-Gaussian not only surpasses SOTA methods by large margins in performance but also enjoys much faster inference speed and cheaper training costs. More specifically, compared with the recent best X-ray NeRF-based method NAF, our X-Gaussian outperforms it by 6.5 dB on average and is $73 \times$ faster in inference while requiring less than 15% training time. When compared with the SOTA RGB NeRF-based method TensorRF, our X-Gaussian is 12.93 dB higher while enjoying $192 \times$ inference speed and $20 \times$ training speed.

To intuitively demonstrate the superiority of our method, we plot the PSNR-minute-fps comparison of different algorithms in Fig. 1. The vertical axis represents the performance in PSNR (dB). The horizontal axis indicates the inference speed in fps. The radius of the circle refers to the training time in minutes. It can be seen that X-Gaussian completely takes up the upper-right corner with the shortest training time, showing its extreme advantages in model efficiency.

Qualitative Results. Figs. 5 and 6 depict the qualitative comparisons of NVS on the scenes of pancreas, chest, foot, and head. It can be observed from the zoomed-in patches that previous NeRF-based algorithms fail to render clear

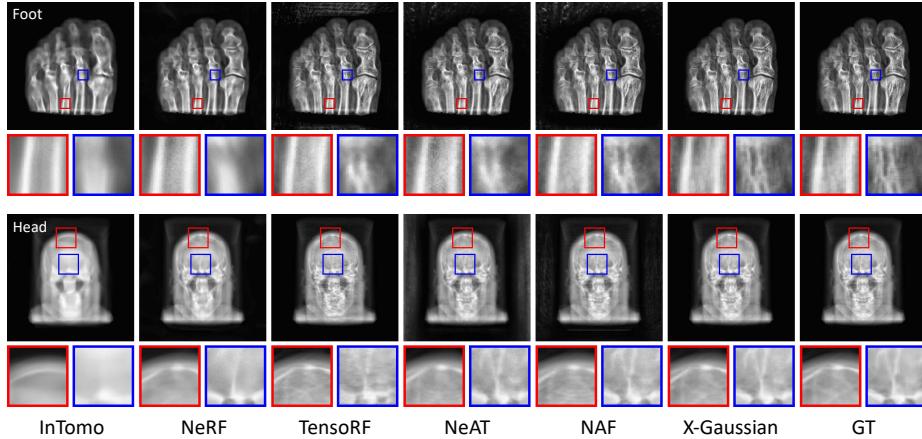


Fig. 6: Qualitative results of novel view synthesis on foot (top) and head (bottom). Our method reconstructs more fine-grained details. Please zoom in for a better view.

novel views. They either introduce undesired artifacts or produce blurry textures such as the toe bones of the foot. In contrast, our method yields visually realistic images by rendering more fine-grained details and clearer structural contents.

4.3 Sparse-View CT Reconstruction

We compare our method with SOTA NeRF-based algorithms on sparse-view CT reconstruction. Since the Gaussian point clouds cannot directly infer the radiodensities of the CT volume, we evaluate different NeRF-based methods and our X-Gaussian by using them to create novel-view projections for three learning-free CT reconstruction methods, including an analytical method (FDK [17]) and two iterative methods (SART [1] and ASD-POCS [45]). Specifically, these three methods reconstruct the CTs from 5 original projections and 95 novel-view projections rendered by different NVS algorithms. The quantitative results are listed in Tab. 2. When only using 5 original views (+ None), FDK, SART, and ASD-POCS achieve 7.41, 17.24, and 17.03 dB in PSNR, respectively. They fail to reconstruct the CT volumes. When employing our X-Gaussian to create novel X-ray projections for FDK, SART, and ASD-POCS, they yield the most significant improvements of 15.19, 13.01, and 13.53 dB in PSNR. These improvements are 1.32, 2.41, and 2.65 dB higher than the improvements of using NAF.

Figs. 7 and 8 show the qualitative results of sparse-view CT reconstruction on the scenes of foot, chest, abdomen, and head. Without using rendered novel-view projections, SART and ASD-POCS fail in reconstructing the CT slices. When using SOTA X-ray NeRF-based methods to create novel views, SART and ASD-POCS produce over-smooth CT slices with blurry structural contents. On the contrary, when using our X-Gaussian to assist SART and ASD-POCS, they can reconstruct much clearer CT slices with more high-frequency textures and fine-grained structural details, such as the vessels in the chest (Fig. 7) and the spine in the head (Fig. 8). These results clearly demonstrate the potential practical values of our method on the sparse-view CT reconstruction task.

Method	+ None + InTomo [58] + NeRF [37] + TensoRF [10] + NeAT [42] + NAF [60]										+ X-Gaussian		
	Metric		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR
FDK		7.41	0.093	20.31	0.498	20.57	0.502	20.61	0.501	20.94	0.511	21.28	0.523
Δ FDK		0	0	12.90	0.405	13.16	0.409	13.20	0.408	13.52	0.418	13.87	0.430
SART		17.24	0.528	26.28	0.859	26.78	0.853	27.06	0.867	27.31	0.869	27.84	0.879
Δ SART		0	0	9.04	0.331	9.54	0.325	9.82	0.339	10.07	0.341	10.60	0.351
ASD-POCS		17.03	0.525	25.44	0.847	26.58	0.857	26.93	0.868	26.95	0.865	27.91	0.880
Δ ASD-POCS		0	0	8.41	0.322	9.55	0.332	9.90	0.343	9.92	0.340	10.88	0.355

Table 2: Results on sparse-view CT reconstruction. NeRF-based methods and our X-Gaussian are used to create novel views for FDK [17], SART [1], and ASD-POCS [45].

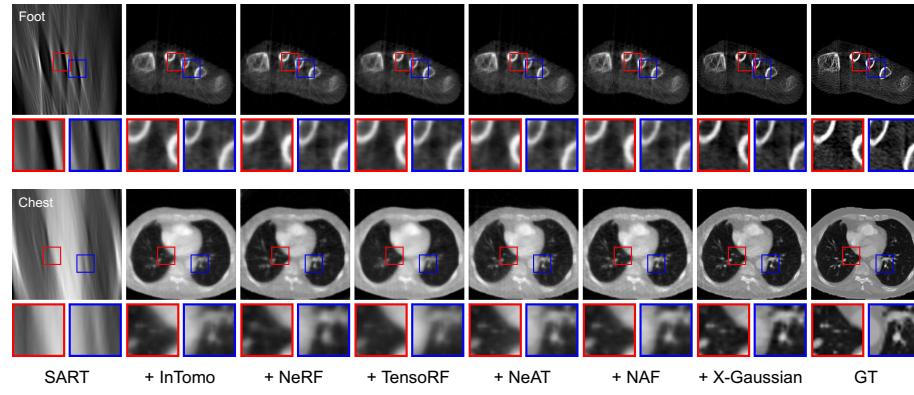


Fig. 7: Visual results of sparse-view CT reconstruction on the scenes of foot and chest. NeRF-based methods and our X-Gaussian are used to create novel views for SART [1].

4.4 Ablation Study

Break-down Ablation. We first conduct a break-down ablation experiment to study the effect of each proposed technique towards higher performance and faster speed. We adopt the original 3DGS [26] as the baseline model and naively average the RGB channels to represent the value of radiation intensity. The results are listed in Tab. 3a. The baseline model yields 37.21 dB PSNR in performance. Its average training time and inference speed are 31 min 38 s and 64 fps, respectively. We can observe from Tab. 3a : (i) When using ACUI to replace the time-consuming SfM [46] algorithm for initialization, the training time is significantly reduced by 34% while the performance yields an improvement of 1.66 dB. This evidence suggests that our ACUI strategy is much faster than the SfM [46] algorithm used in the original 3DGS and can compute more accurate camera calibration parameters for 3D Gaussians and subsequent rendering. (ii) Then we apply our radiative Gaussian point cloud model equipped with the proposed Differentiable Radiative Rasterization (DRR) to replace the original RGB Gaussian point cloud model and its RGB rasterization. As analyzed in Sec. 3.1 and compared in Fig. 4, the anisotropic spherical harmonics (SH) are not suitable for X-ray imaging because X-ray imaging based on penetration is isotropic. In contrast, our radiative Gaussian point cloud model can better fit the

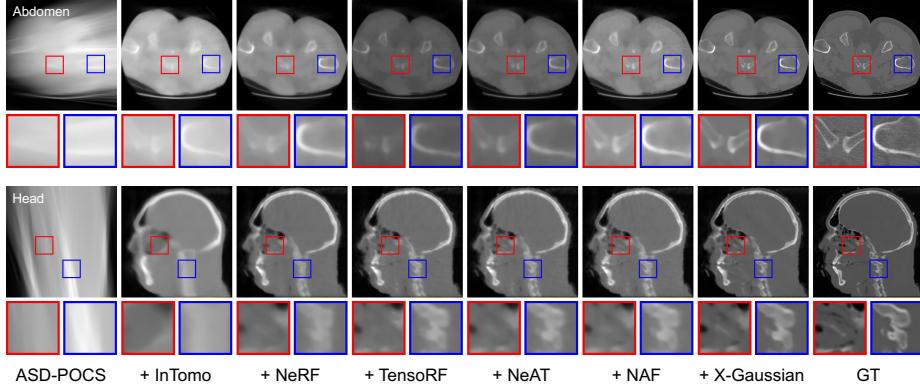


Fig. 8: Visual results of sparse-view CT reconstruction on abdomen and head. NeRF-based methods and X-Gaussian are used to create novel views for ASD-POCS [45].

view-independent radiation intensity in 3D space. Therefore, the performance is significantly improved by 4.53 dB in PSNR. Besides, removing the computation related to the view direction from the forward and backward processes of rasterization can further accelerate the training and inference speed. Thus, the training time is reduced by 54.10% and the inference speed is $2.1\times$ faster.

Initialization of Point Position. We compare different initialization strategies for the center positions of Gaussian point clouds including random, spherical, FDK [17], and cubic initialization. To be specific, Random initialization means randomly sampling points within the scanned area in 3D space. Spherical initialization uniformly samples point positions within a sphere that can completely enclose the scanned object. FDK [17] initialization adopts the FDK algorithm to back-project the given projections into 3D point positions. Cuboid initialization is our ACUI. Please note that we keep the computed intrinsic and extrinsic matrices the same for fair comparison between different strategies. The results are reported in Tab. 3b. FDK initialization slightly outperforms our ACUI by 0.066 dB but its training time is $2.59\times$ longer and its inference speed is 55 fps slower than ACUI. This is because the back-projection in FDK is time-consuming and initializes redundant points. The random and spherical initialization strategies yield lower PSNR and slower speed than cubic initialization. To achieve a better trade-off, we adopt the cubic initialization, *i.e.*, ACUI, which enjoys good performance, the cheapest training cost, and the fastest inference speed.

Parameter Analysis. We conduct parameter analysis of the number of features N_f and the sampling interval d . The results are shown in Tab. 3c and Tab. 3d.

In Tab. 3c, (i) When increasing N_f , the performance gradually improves but the magnitude of the improvement decreases. In particular, $N_f = 32$ achieves the best results of 43.42 dB in PSNR. $N_f = 16$ achieves on-par results with $N_f = 32$, only 0.013 dB lower. (ii) We notice that the training time and inference speed do not change monotonically. This is because the Gaussian point clouds with various feature dimensions have different representing ability and

Method	3DGS [26]	+ ACUI	+ DRR	Method	Random	Spherical	FDK [17]	Cubic
PSNR	37.213	38.872	43.404	PSNR	41.329	42.837	43.470	43.404
SSIM	0.9813	0.9871	0.9993	SSIM	0.9942	0.9988	0.9993	0.9993
Train time (s)	1898	1172	538	Train time (s)	601	575	1394	538
Infer speed (fps)	64	72	148	Infer speed (fps)	112	136	93	148

(a) Break-down ablation study						
Number	1	2	4	8	16	32
PSNR	38.818	40.205	42.130	42.868	43.404	43.417
SSIM	0.9840	0.9931	0.9983	0.9991	0.9993	0.9993
Train time (s)	511	525	521	553	538	752
Speed (fps)	153	152	158	127	148	101

(b) Ablation of point position initialization						
Interval	1	2	4	8	16	32
PSNR	42.853	42.979	43.215	43.404	43.311	43.294
SSIM	0.9989	0.9990	0.9992	0.9993	0.9992	0.9992
Train time (s)	785	593	545	538	534	566
Speed (fps)	86	94	114	148	135	97

(c) Analysis of the number of features N_f						
Number	1	2	4	8	16	32
PSNR	38.818	40.205	42.130	42.868	43.404	43.417
SSIM	0.9840	0.9931	0.9983	0.9991	0.9993	0.9993
Train time (s)	511	525	521	553	538	752
Speed (fps)	153	152	158	127	148	101

(d) Analysis of the initialized interval d						
Interval	1	2	4	8	16	32
PSNR	42.853	42.979	43.215	43.404	43.311	43.294
SSIM	0.9989	0.9990	0.9992	0.9993	0.9992	0.9992
Train time (s)	785	593	545	538	534	566
Speed (fps)	86	94	114	148	135	97

Table 3: Ablation study. PSNR, SSIM, training time, and inference speed are reported.

computational complexity. The number of final 3D Gaussians after training also varies substantially. When $N_f = 16$, the training time reaches a local minimum and the inference speed is at its local maximum. Hence, we eventually adopt $N_f = 16$ to reach a more optimal balance between performance and speed.

In Tab. 3d, the best performance, the cheapest memory cost, and the fastest inference speed are achieved at $d = 8$. The training time (538 s) at $d = 8$ is almost the same as the shortest one (534 s) at $d = 16$. Thus, we set d to 8.

Convergence Analysis. We conduct two visual analyses to compare the convergence between our X-Gaussian and original 3DGS [26] in Fig. 2, and between X-Gaussian and the SOTA X-ray NeRF-based method NAF [60] in Fig. 9.

Specifically, we adopt the same ACUI strategy for the original 3DGS to focus on comparing the Gaussian point cloud model and the rasterization. For fairness, we train 3DGS and our X-Gaussian on the scene of foot with the same settings and visualize the positions and opacities of Gaussian point clouds at the 100-th, 1000-th, 5000-th, 10000-th, and 20000-th iterations of the training process in Fig. 2. We also visualize the CT volume of foot as a reference. Please note that the CT volume is not the ground truth of point clouds. As can be seen that the original 3DGS with RGB rasterization converges slowly and suffers from more noisy point clouds. Plus, the final trained model of 3DGS at the 20000-th iteration contains more redundant Gaussians that are unnecessary to represent the 3D structure of the foot, which reduces the model’s inference speed. In contrast, our X-Gaussian equipped with the proposed DRR shows faster and better convergence. In particular, as early as the 1000-th iteration, our radiative Gaussian point clouds have essentially formed the basic shape of the foot. Besides, the final trained X-Gaussian at the 20000-th iteration can better represent the 3D geometry and more accurate structural contents than the original 3DGS.

Additionally, in Fig. 9, we visualize the rendered novel projections with the same angle ϕ of NAF and our X-Gaussian at 20s, 60s, and 180s of the training process on the scene of chest. NAF produces blurry images with severe noises on the background regions within the first three minutes of the training. In contrast, our X-Gaussian can reconstruct clearer structural details like the ribs and blood vessels with cleaner background of the chest at the first minute of the training.

The two visual analyses show the convergence advantages of our X-Gaussian.

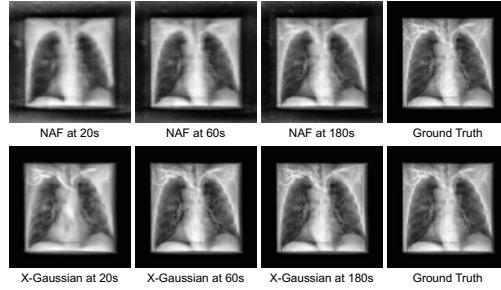


Fig. 9: Convergence analysis of NAF [60] *vs.* our X-Gaussian. We visualize the rendered projections at 20s, 60s, and 180s of training. Our X-Gaussian shows faster and better convergence.

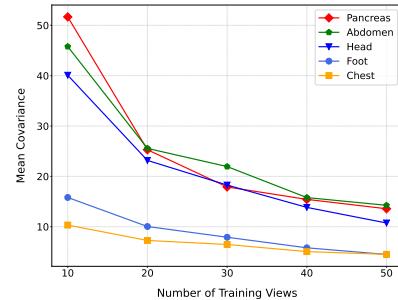


Fig. 10: Analysis of covariance. The mean covariance of 3D Gaussians on different scenes decreases when the number of training views increases.

Analysis of Covariance. We study how the shape of the 3D Gaussian point cloud changes with the number of training views in Fig. 10. As the number of training views increases, the mean covariance of 3D Gaussians that control the size of Gaussian point clouds decreases. This indicates that the 3D Gaussian point clouds gradually change from coarse to fine, thereby being more capable of representing fine-grained structures, such as a small tumor in the abdomen.

5 Limitation

Compared with NeRF-based methods, our X-Gaussian is more complex and harder to follow because it requires more foundational background knowledge in computer graphics and 3D vision. Many technical details of our X-Gaussian are implemented by CUDA instead of Pytorch. CUDA based on C++ is more difficult to debug and less interpretable than Pytorch which relies on Python.

6 Conclusion

In this paper, we propose the first 3DGS-based framework, X-Gaussian, for X-ray novel view synthesis. Firstly, we design a radiative Gaussian point cloud model based on X-ray imaging properties. This model excludes the influence of view direction when fitting radiation intensity. For this model, we develop a GPU-friendly differentiable radiative rasterization CUDA kernel that renders projections at a faster speed than RGB rasterization. Secondly, we customize an initialization strategy, ACUI, that does not need to execute the SfM algorithm. Instead, ACUI uses the parameters of X-ray scanner to compute the extrinsic and intrinsic matrices, and then uniformly samples center points for 3D Gaussians within a cuboid enclosing the scanned object. Experiments demonstrate that our X-Gaussian significantly outperforms SOTA methods by over 6.5 dB while enjoying 73 \times faster inference speed and only requiring 15% of training time.

Acknowledgements: This work was supported by the Lustgarten Foundation for Pancreatic Cancer Research and the Patrick J. McGovern Foundation Award.

References

1. Andersen, A.H., Kak, A.C.: Simultaneous algebraic reconstruction technique (sart): a superior implementation of the art algorithm. *Ultrasonic imaging* (1984) [10](#), [11](#)
2. Armato III, S.G., McLennan, G., Bidaut, L., McNitt-Gray, M.F., Meyer, C.R., Reeves, A.P., Zhao, B., Aberle, D.R., Henschke, C.I., Hoffman, E.A., et al.: The lung image database consortium (lidc) and image database resource initiative (idri): a completed reference database of lung nodules on ct scans. *Medical physics* (2011) [8](#)
3. Barron, J.T., Mildenhall, B., Tancik, M., Hedman, P., Martin-Brualla, R., Srinivasan, P.P.: Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: *ICCV* (2021) [3](#)
4. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: *CVPR* (2022) [3](#)
5. Barron, J.T., Mildenhall, B., Verbin, D., Srinivasan, P.P., Hedman, P.: Zip-nerf: Anti-aliased grid-based neural radiance fields. In: *ICCV* (2023) [3](#)
6. Biguri, A., Dosanjh, M., Hancock, S., Soleimani, M.: Tigre: a matlab-gpu toolbox for cbct image reconstruction. *Biomedical Physics & Engineering Express* (2016) [8](#)
7. Boone, J., Shah, N., Nelson, T.: A comprehensive analysis of coefficients for pendant-geometry cone-beam breast computed tomography. *Medical physics* (2004) [1](#)
8. Boone, J.M., Nelson, T.R., Lindfors, K.K., Seibert, J.A.: Dedicated breast ct: radiation dose and image quality evaluation. *Radiology* (2001) [1](#)
9. Cai, Y., Wang, J., Yuille, A., Zhou, Z., Wang, A.: Structure-aware sparse-view x-ray 3d reconstruction. In: *CVPR* (2023) [3](#)
10. Chen, A., Xu, Z., Geiger, A., Yu, J., Su, H.: Tensorf: Tensorial radiance fields. In: *ECCV* (2022) [3](#), [8](#), [9](#), [11](#)
11. Chen, B., Ning, R.: Cone-beam volume ct breast imaging: Feasibility study. *Medical physics* (2002) [1](#)
12. Chen, Z., Funkhouser, T., Hedman, P., Tagliasacchi, A.: Mobilenerf: Exploiting the polygon rasterization pipeline for efficient neural field rendering on mobile architectures. In: *CVPR* (2023) [3](#)
13. Cormack, A.M.: Representation of a function by its line integrals, with some radiological applications. *Journal of applied physics* (1963) [1](#)
14. Cormack, A.M.: Representation of a function by its line integrals, with some radiological applications. ii. *Journal of Applied Physics* (1964) [1](#)
15. Corona-Figueroa, A., Frawley, J., Bond-Taylor, S., Bethapudi, S., Shum, H.P., Willcocks, C.G.: Mednerf: Medical neural radiance fields for reconstructing 3d-aware ct-projections from a single x-ray. In: *International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)* (2022) [3](#)
16. Elbakri, I.A., Fessler, J.A.: Segmentation-free statistical image reconstruction for polyenergetic x-ray computed tomography with experimental validation. *Physics in Medicine & Biology* (2003) [1](#)
17. Feldkamp, L.A., Davis, L.C., Kress, J.W.: Practical cone-beam algorithm. *Josa a* (1984) [10](#), [11](#), [12](#), [13](#)
18. Guide, D.: Cuda c programming guide. NVIDIA (2013) [8](#)
19. Hounsfield, G.N.: Computerized transverse axial scanning (tomography): Part 1. description of system. *The British journal of radiology* (1973) [1](#)
20. Hounsfield, G.N.: Computed medical imaging. *Science* (1980) [1](#)

21. Hu, S., Liu, Z.: Gauhuman: Articulated gaussian splatting from monocular human videos. arXiv preprint arXiv: (2023) 4
22. Hu, T., Liu, S., Chen, Y., Shen, T., Jia, J.: Efficientnerf efficient neural radiance fields. In: CVPR (2023) 3
23. Hu, W., Wang, Y., Ma, L., Yang, B., Gao, L., Liu, X., Ma, Y.: Tri-miprf: Tri-mip representation for efficient anti-aliasing neural radiance fields. In: ICCV (2023) 3
24. Jiang, Y., Tu, J., Liu, Y., Gao, X., Long, X., Wang, W., Ma, Y.: Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces. arXiv preprint arXiv:2311.17977 (2023) 4
25. Keetha, N., Karhade, J., Jatavallabhula, K.M., Yang, G., Scherer, S., Ramanan, D., Luiten, J.: Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. arXiv preprint arXiv:2312.02126 (2023) 4
26. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics (2023) 2, 4, 5, 6, 7, 11, 13
27. Kingma, D.P., Ba, J.L.: Adam: A method for stochastic optimization. In: ICLR (2015) 8
28. Klacansky, P.: Scientific visualization datasets (2022), <https://klacansky.com/open-scivis-datasets/> 8
29. Kocabas, M., Chang, J.H.R., Gabriel, J., Tuzel, O., Ranjan, A.: Hugs: Human gaussian splats. arXiv preprint arXiv:2311.17910 (2023) 4
30. Kopanas, G., Philip, J., Leimkühler, T., Drettakis, G.: Point-based neural rendering with per-view optimization. In: Computer Graphics Forum (2021) 6
31. Li, R., Gao, H., Tancik, M., Kanazawa, A.: Nerface: Efficient sampling accelerates nerfs. In: ICCV (2023) 3
32. Liang, Y., Yang, X., Lin, J., Li, H., Xu, X., Chen, Y.: Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. arXiv preprint arXiv:2311.11284 (2023) 4
33. Liang, Z., Zhang, Q., Feng, Y., Shan, Y., Jia, K.: Gs-ir: 3d gaussian splatting for inverse rendering. arXiv preprint arXiv:2311.16473 (2023) 4
34. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. arXiv preprint arXiv:2308.09713 (2023) 4
35. Manglos, S.H., Gagne, G.M., Krol, A., Thomas, F.D., Narayanaswamy, R.: Transmission maximum-likelihood reconstruction with ordered subsets for cone beam ct. Physics in Medicine & Biology (1995) 1
36. Matsuki, H., Murai, R., Kelly, P.H., Davison, A.J.: Gaussian splatting slam. arXiv preprint arXiv:2312.06741 (2023) 4
37. Mildenhall, B., Srinivasan, P., Tancik, M., Barron, J., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020) 2, 3, 8, 9, 11
38. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM ToG (2022) 3
39. Pan, J., Zhou, T., Han, Y., Jiang, M., et al.: Variable weighted ordered subset image reconstruction algorithm. International Journal of Biomedical Imaging (2006) 1
40. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. In: NeurIPS (2019) 8
41. Reiser, C., Szeliski, R., Verbin, D., Srinivasan, P., Mildenhall, B., Geiger, A., Barron, J., Hedman, P.: Merf: Memory-efficient radiance fields for real-time view synthesis in unbounded scenes. TOG (2023) 3

42. Rückert, D., Wang, Y., Li, R., Idoughi, R., Heidrich, W.: Neat: Neural adaptive tomography. TOG (2022) **8**, **9**, **11**
43. Sauer, K., Bouman, C.: A local update strategy for iterative reconstruction from projections. TIP (1993) **1**
44. Scarfe, W.C., Farman, A.G., Sukovic, P., et al.: Clinical applications of cone-beam computed tomography in dental practice. Journal-Canadian Dental Association (2006) **1**
45. Sidky, E.Y., Pan, X.: Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. Physics in Medicine & Biology (2008) **10**, **11**, **12**
46. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3d. In: SIGGRAPH (2006) **2**, **7**, **11**
47. Tang, J., Ren, J., Zhou, H., Liu, Z., Zeng, G.: Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. arXiv preprint arXiv:2309.16653 (2023) **4**
48. Verbin, D., Hedman, P., Mildenhall, B., Zickler, T., Barron, J.T., Srinivasan, P.P.: Ref-nerf: Structured view-dependent appearance for neural radiance fields. In: CVPR (2022) **3**
49. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: from error visibility to structural similarity. TIP (2004) **9**
50. Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. arXiv preprint arXiv:2310.08528 (2023) **4**
51. Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., Jiang, C.: Physgaussian: Physics-integrated 3d gaussians for generative dynamics. arXiv preprint arXiv:2311.12198 (2023) **4**
52. Yan, C., Qu, D., Wang, D., Xu, D., Wang, Z., Zhao, B., Li, X.: Gs-slam: Dense visual slam with 3d gaussian splatting. arXiv preprint arXiv:2311.11700 (2023) **4**
53. Yang, Z., Yang, H., Pan, Z., Zhu, X., Zhang, L.: Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. arXiv preprint arXiv:2310.10642 (2023) **4**
54. Yariv, L., Hedman, P., Reiser, C., Verbin, D., Srinivasan, P.P., Szeliski, R., Barron, J.T., Mildenhall, B.: Bakedsdf: Meshing neural sdbs for real-time view synthesis. In: SIGGRAPH (2023) **3**
55. Yi, T., Fang, J., Wu, G., Xie, L., Zhang, X., Liu, W., Tian, Q., Wang, X.: Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. arXiv preprint arXiv:2310.08529 (2023) **4**
56. Yu, L., Zou, Y., Sidky, E.Y., Pelizzari, C.A., Munro, P., Pan, X.: Region of interest reconstruction from truncated data in circular cone-beam ct. TMI (2006) **1**
57. Yugay, V., Li, Y., Gevers, T., Oswald, M.R.: Gaussian-slam: Photo-realistic dense slam with gaussian splatting. arXiv preprint arXiv:2312.10070 (2023) **4**
58. Zang, G., Idoughi, R., Li, R., Wonka, P., Heidrich, W.: Intratomo: self-supervised learning-based tomography via sinogram synthesis and prediction. In: CVPR (2021) **3**, **8**, **9**, **11**
59. Zbijewski, W., Defrise, M., Viergever, M.A., Beekman, F.J.: Statistical reconstruction for x-ray ct systems with non-continuous detectors. Physics in Medicine & Biology (2006) **1**
60. Zha, R., Zhang, Y., Li, H.: Naf: neural attenuation fields for sparse-view cbct reconstruction. In: MICCAI (2022) **2**, **3**, **8**, **9**, **11**, **13**, **14**
61. Zwicker, M., Pfister, H., Van Baar, J., Gross, M.: Ewa volume splatting. In: Proceedings Visualization, 2001. VIS'01. IEEE (2001) **6**