BACHELOR OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

Orthographic Gaussian Splatting from Axially Stacked Radiographs via SfM-Guided Novel View Synthesis

Tasrefee Mahmood Ridun
200042150
MD Mushfiqur Rahman Mushfique
200042160
Kazi Jawadul Islam Jishan
200042165

Department of Computer Science and Engineering
Islamic University of Technology
November, 2025

Orthographic Gaussian Splatting from Axially Stacked Radiographs via SfM-Guided Novel View Synthesis

Tasrefee Mahmood Ridun
200042150
MD Mushfiqur Rahman Mushfique
200042160
Kazi Jawadul Islam Jishan
200042165

Department of Computer Science and Engineering
Islamic University of Technology
November, 2025

Declaration of Candidate

This is to certify that the work presented in this thesis is the outcome of the analysis and experiments carried out by **Tasrefee Mahmood Ridun**, **MD Mushfiqur Rahman Mushfique**, and **Kazi Jawadul Islam Jishan** under the supervision of **Tareque Mohmud Chowdhury**, Assistant Professor, Department of Computer Science and Engineering, Islamic University of Technology (IUT), OIC, Islamic University of Technology, Dhaka, Bangladesh. It is also declared that neither this thesis nor any part of it has been submitted anywhere else for any degree or diploma. Information derived from the published and unpublished work of others have been acknowledged in the text and a list of references is given.

Tareque Mohmud Chowdhury
Assistant Professor
Department of Computer Science and Engineering, Islamic University of Technology (IUT), OIC
Islamic University of Technology (IUT)
Date: November 28, 2025

Tasrefee Mahmood Ridun
Student ID: 200042150
Date: November 28, 2025

MD Mushfiqur Rahman Mushfique
Student ID: 200042160
Date: November 28, 2025

Kazi Jawadul Islam Jishan
Student ID: 200042165
Date: November 28, 2025

Dedicated to the people of Palestine.

Contents

List of Tables

Acknowledgement

Acknowledgments

We are profoundly grateful to our supervisor, Tareque Mohmud Chowdhury, for his endless patience, insightful critique, and His expertise and encouragement were instrumental in the completion of this thesis.

Finally, to our parents, whose love and support have been a constant source of strength. Thank you for always being there.

To all of you, we extend our heartfelt appreciation.

Abstract

We present a Gaussian-splatting pipeline tailored to axially stacked radiographs that breaks with the perspective and alpha-blending assumptions of standard 3DGS. Clinical CT data are orthographic and exhibit little cross-slice feature continuity, causing classical SfM and vanilla splat renderers to fail (edge-biased splats, central collapse). Our method first converts ordered CT slices into a metric volume, then renders **radiographic** projections that obey Beer–Lambert attenuation to synthesize multi-view images with controlled overlap. From these, we obtain poses and train a **rectified radiative Gaussian** model that replaces alpha compositing with additive X-ray accumulation and includes a density-rectification term so each Gaussian's parameter encodes true 3D density rather than view-integrated mass. We initialize Gaussians directly from the slice volume with a lightweight sampler that seeds positions, scales, and densities, and stabilize optimization with gentle TV-on-voxel readouts and adaptive clone/split densification. On a 1.8k-slice brain CT (HiP-CT family), using 1.2k synthetic projections, our system reconstructs coherent anatomy and produces faithful novel projections after 45 minutes of training, visibly restoring central structures and suppressing edge-only artifacts compared to a cinematic-GS baseline. The approach is code-practical (no reliance on FDK/TIGRE), data-efficient, and compatible with DICOM geometry, offering a reproducible path to fast, radiography-consistent 3D reconstructions from orthographic medical stacks. We discuss remaining limits (pose accuracy, scatter, regularization strength) and outline ablations and metrics (L1/SSIM on held-out views, orthogonal-slice fidelity) to guide future clinical validation.

Chapter 1

Introduction

Neural view synthesis has advanced rapidly from implicit neural radiance fields (NeRFs) to explicit, rasterization-friendly point/ellipsoid primitives, enabling real-time rendering with high fidelity. In particular, *3D Gaussian Splatting* (3DGS) represents scenes as anisotropic Gaussians optimized from posed images and rendered with a fast, visibility-aware splatter—achieving real-time performance while maintaining state-of-the-art quality [**?**]. Subsequent work has addressed aliasing via mip-style prefiltering [**?**] and extended splatting to dynamics (4DGS) [**?**], while the NeRF literature contributed anti-aliasing in unbounded scenes (Mip-NeRF 360) [**?**] and the foundational implicit-field formulation [**?**]. These developments position Gaussian splatting as a practical bridge between physically based volumetric rendering and GPU-efficient rasterization.

Medical imaging presents distinct demands. Clinical CT/X-ray projections obey transmission physics (Beer–Lambert attenuation) rather than the front-to-back alpha blending assumed in conventional 3DGS, and data often arrive as orthographic axial stacks with negligible cross-slice feature overlap, frustrating standard structure-from-motion (SfM) pipelines [**?** **?**]. Emerging *radiative* Gaussian variants explicitly adapt the rasterizer and loss to X-ray formation—e.g., X-Gaussian for efficient X-ray novel-view synthesis [**?**] and R^2-Gaussian for tomographic reconstruction with a bias-corrected integral [**?**]. In anatomy visualization, cinematic and layered GS approaches compress path-traced medical volumes into interactive splat models [**?** **?**]. Within this context, we study a splatting-based pipeline tailored to orthographic CT stacks, focusing on pose acquisition from synthetic radiographic views, physics-consistent compositing, and stable initialization—aimed at accurate novel-projection synthesis and interactive 3D exploration of volumetric anatomy. We ground our design in the rendering equation

[**?**], radiographic physics [**?**], and established reconstruction baselines [**? ?**].

1.1 What Sections Are Included in This Chapter

Section 1.2 reviews Gaussian splatting fundamentals. Section 1.3 states the motivation and scope, emphasizing radiographic view synthesis from orthographic CT stacks. Section 1.4 formulates research questions and deliverables. Section 1.5 provides a formal problem statement. Section 1.6 outlines challenges. Section 1.7 summarizes our contributions relative to the literature.

1.2 Introductory Information

Gaussian primitive. A 3D Gaussian is parameterized by mean $\boldsymbol{\mu} \in \mathbb{R}^3$, covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$ (often via a scale–rotation factorization), color $\mathbf{c} \in [0,1]^3$, and per-primitive opacity $\alpha \in [0,1]$. Under a pinhole camera π, splatting rasterizes the screen-space footprint of each Gaussian with a visibility-aware, order-independent blend that approximates volumetric rendering [**? ?**].

Training objective. Given posed images $\{I_k\}$ with cameras $\{\pi_k\}$ and parameters $\Theta = \{\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \mathbf{c}_i, \alpha_i\}_{i=1}^N$, we minimize a photometric loss plus regularizers:

$$\mathcal{L}(\Theta) = \sum_k \ell\big(\mathcal{R}(\Theta;\pi_k),\, I_k\big) + \lambda_{\text{reg}}\mathcal{R}_{\text{aux}}(\Theta), \tag{1.1}$$

where $\mathcal{R}$ is the differentiable rasterizer; ℓ typically includes ℓ_1/ℓ_2 and perceptual penalties; evaluation uses PSNR, SSIM, and LPIPS [**? ?**]. Density control (prune/-clone/split) and anisotropic covariance optimization improve coverage and sharpness [**? ?**].

Radiative (X-ray) rendering. For transmission imaging with linear attenuation $\mu(\mathbf{x})$, pixel intensity follows Beer–Lambert:

$$I = I_0 \exp\!\left(-\int_{\text{ray}} \mu(\mathbf{x})\,\mathrm{d}l\right). \tag{1.2}$$

Radiative Gaussian splatting replaces alpha compositing with additive line integrals or their closed-form approximations through Gaussian kernels, aligning supervision with physics [**? ? ?**].

1.3 Motivation and Scope

Orthographic CT stacks (e.g., DICOM/HiP-CT) are ubiquitous in clinical and research workflows but are poorly matched to perspective-SfM assumptions and to alpha-blended 3DGS. A domain-appropriate pipeline can (i) recover camera geometry for orthographic slices via *synthetic radiographs* with controlled overlap, (ii) train a *radiative* Gaussian model consistent with X-ray physics, and (iii) preserve anatomical fidelity while enabling interactive rendering on commodity GPUs. We scope the study to *static* anatomy from axial stacks, posed or synthesized projections, and quantitative view-space metrics with slice-space sanity checks. Broader topics—artifact correction (scatter, beam hardening), full clinical validation, or regulatory concerns—are noted but out of scope for this chapter. Representative prior art includes 3DGS [**?**], mip-prefiltering for aliasing [**? ?**], radiative splatting for X-rays [**? ?**], and anatomy-focused GS organization [**? ?**].

1.4 Scope of Research

We investigate the following questions:

1. **Pose acquisition from axial stacks.** Can synthetic radiographic projections with designed overlap enable robust SfM pose recovery for orthographic CT data, avoiding manual pose engineering? We compare COLMAP-based pipelines using generated views versus naive slice-to-slice matching [**?**].

2. **Radiative compositing in GS.** Does replacing alpha blending with a Beer–Lambert-consistent rasterizer improve novel X-ray projection fidelity and geometric coherence relative to vanilla 3DGS? [**? ? ?**].

3. **Initialization and density control.** How do point-cloud seeds derived directly from slices versus uniform priors affect convergence, pruning, and central-structure preservation? We draw on density control strategies from 3DGS and radiative initializations [**? ?**].

4. **Quality–speed–memory trade-offs.** What are the runtime and memory costs to achieve useful fidelity, and how do mip-style anti-aliasing and multi-layer organization affect this trade-off? [**? ?**].

Deliverables. (i) A reproducible pipeline for synthetic radiograph generation, pose estimation, and radiative GS training; (ii) ablations on initialization, rasterization, and anti-aliasing; (iii) quantitative and qualitative evaluation (PSNR/SSIM/LPIPS and

slice-space checks). Metrics and datasets are selected with attention to licensing and privacy where applicable [**? ? ? ?**].

1.5 Problem Statement

Let $\mathcal{V} = \{S_z\}_{z=1}^{Z}$ denote an orthographic CT stack resampled to metric spacing. We synthesize a set of radiographic views $\{(I_k, \pi_k)\}_{k=1}^{K}$ with known extrinsics/intrinsics by rendering line integrals through $\mathcal{V}$. We seek parameters Θ of a radiative Gaussian field $\mathcal{G}$ such that its rendered projections match the observed/synthetic images under Beer–Lambert physics:

$$\min_{\Theta} \sum_{k=1}^{K} \underbrace{\ell\big(\mathcal{R}_{\text{rad}}(\mathcal{G}(\Theta); \pi_k),\, I_k\big)}_{\text{image loss}} + \lambda\, \mathcal{R}_{\text{dens}}(\Theta), \tag{1.3}$$

subject to resource constraints (memory $< M$, render FPS $\geq f$) and pose set $\{\pi_k\}$ obtained via SfM on the synthetic set. The renderer $\mathcal{R}_{\text{rad}}$ approximates $\int \mu(\mathbf{x})\, \mathrm{d}l$ per pixel using Gaussian kernels, avoiding standard alpha compositing [**? ? ?**]. Performance is evaluated by PSNR, SSIM, and LPIPS on held-out projections, plus qualitative slice-space overlays [**? ?**].

1.6 Research Challenges

- **Pose from orthographic data.** Adjacent axial slices share little texture; naive feature matching fails. Generating *synthetic* radiographs with controlled parallax and overlap is required for reliable SfM [**?**].
- **Physics-consistent compositing.** Alpha blending in 3DGS violates X-ray transmission physics, biasing densities; radiative line-integral splatting is needed [**? ? ?**].
- **Initialization and coverage.** Uniform/random seeds can collapse central structures; slice-aware seeding and adaptive density control are key [**? ?**].
- **Aliasing and resolution.** Large zoom/scale changes and thin structures necessitate alias-free prefiltering and possibly multi-layer organization [**? ?**].
- **Quality–speed–memory trade-offs.** Radiative kernels and many Gaussians can bloat memory; careful pruning, mip prefiltering, and layer compression are required for interactive rates [**? ? ?**].

- **Evaluation and ethics.** Perceptual (LPIPS) and structural (SSIM) metrics complement PSNR; dataset licensing (e.g., DICOM/HiP-CT) and privacy must be respected [**? ?**].

1.7 Contributions

1. **SfM-from-synthetics for orthographic CT.** A method to convert axial stacks into *synthetic radiographic views* with designed overlap and coverage, enabling robust COLMAP pose estimation without real multi-view acquisitions. Addresses pose failure modes on slice data and establishes a reproducible route to camera geometry [**?**].
2. **Radiative Gaussian rasterization for projections.** A Beer–Lambert-consistent splatting objective and renderer for novel X-ray views, reducing density bias observed with alpha compositing and improving fidelity on thin/anatomical structures [**? ? ?**].
3. **Slice-aware initialization and density control.** A lightweight point-cloud seeding strategy (from the slice volume) combined with adaptive prune/clone/split to preserve central structures and accelerate convergence, extending 3DGS practices to the radiographic regime [**? ?**].
4. **Comprehensive ablations and metrics.** Ablations on initialization, compositing model, and anti-aliasing (mip filters), with evaluation on held-out projections via PSNR/SSIM/LPIPS and qualitative slice-space overlays [**? ? ?**].
5. **Reproducible pipeline and release.** A code-practical system compatible with DICOM geometry and general datasets (LLFF/Tanks-and-Temples for sanity checks) [**? ?**].

Chapter 2

Gaussian Splat

A Gaussian splat is a technique used in 3D rendering and visualization, where 3D data points are represented not as discrete points or voxels, but as blurred or smooth "splats" that have a Gaussian distribution. Essentially, it's a method of approximating a surface or volume by projecting Gaussian functions into 2D space, creating a soft, circular blur-like effect for each data point. These splats can represent either the shape or color of a surface, contributing to the overall appearance when viewed from different perspectives.

2.1 What is a Gaussian Splat?

Main Goal

The primary objective is to enable **real-time, high-quality rendering** (e.g., 30+ FPS at 1080p) of 3D scenes from multi-view captures, without requiring extensive training times typically associated with NeRF models.

Approach

The method involves the following innovations:

- A **novel scene representation**: A cloud of **anisotropic 3D Gaussians**.
- A **differentiable rasterizer**: A fast, visibility-aware, tile-based rendering engine optimized for GPU performance.

This approach combines the rendering efficiency of GPU-optimized game engines with the realism typically associated with NeRF models.

Part 1: 3D Gaussian Representation

Instead of using voxels, MLPs, or dense point clouds, the authors utilize:

- **3D Gaussians**, where each Gaussian is characterized by:
  - A 3D **position**.
  - A **covariance matrix** (which controls its shape, akin to a flexible ellipsoid).
  - **Opacity** (alpha).
  - **Spherical Harmonics (SH) coefficients** for color, which aid in view-dependent lighting.

Anisotropic Gaussians are more versatile than simple spherical blobs — they stretch and compress to model geometry such as thin rails or leaves.

Notably, this approach eliminates the need for normals, bypassing a typically complex geometric step.

Part 2: Optimization & Density Control

Given the collection of Gaussians, optimization is required. This process can be likened to sculpting a statue from raw material:

- **Stochastic Gradient Descent (SGD)**:
  - Optimizes each Gaussian's position, shape, color, and opacity.
  - The loss function combines **L1** error and **D-SSIM** to maintain sharpness and structural fidelity.
- **Adaptive Densification**:
  - Areas of the scene that are **under-reconstructed** are enhanced by cloning Gaussians.
  - Areas that are **over-reconstructed** are refined by splitting large Gaussians into smaller ones.
  - Gaussians with minimal contribution (e.g., transparent or overly large Gaussians) are **pruned**.

- This is a dynamic optimization process — Gaussians are created, adjusted, and removed to best fit the scene.

Part 3: Fast Differentiable Tile Rasterizer

The rendering process is optimized through the following steps:

- The screen is divided into **16×16 tiles**.
- For each tile, the Gaussians that **project into it** are gathered and sorted by depth.
- A **GPU radix sort** ensures that proper visibility (front-to-back) is maintained.
- **Alpha-blending** is applied, similar to the method used for semi-transparent objects in real-time graphics.

The key advantages of this approach include:

- It is **differentiable**, allowing it to work seamlessly with backpropagation during training.
- There is no upper limit on the number of Gaussians that can influence a given pixel, making it suitable for complex scenes.

Part 4: Results

When compared to **Mip-NeRF360**, **InstantNGP**, and **Plenoxels**, the system demonstrates:

- Comparable or superior **visual quality**.
- Rendering speeds of up to **135 FPS**.
- Training times reduced to **minutes**, as opposed to hours.

Part 5: Limitations

Despite its successes, the system has certain limitations:

- **Artifacts** in areas that were poorly captured or are not visible in the dataset.
- **Splotchy or elongated Gaussians** in certain situations.

- **Popping** effects when large Gaussians shift visibility order.
- **High memory usage** during training, though this can be optimized.

In Summary

Gaussian Splatting presents a hybrid approach:

- Combining **explicit** techniques (such as point clouds, which provide speed).
- With the **smoothness and differentiability** of NeRFs (which excel at learning from images).

This represents one of the closest approximations to real-time, high-fidelity 3D graphics generated from photographs — without the need for manual modeling or extensive pre-processing.

2.2 3D Gaussian Representation

This section provides a detailed, rigorous explanation of the components that define a 3D Gaussian. Each component plays a critical role in how these Gaussians cooperate to reconstruct a scene from multi-view images.

---

1. Mean – The Position in 3D Space

The **mean** represents the simplest and most fundamental aspect of a 3D Gaussian. It specifies the **location** of the Gaussian in 3D space.

One can conceptualize the mean as the **center point** of a soft, flexible shape. For example, if reconstructing a chair, the Gaussian may be placed near one of its legs. Each Gaussian is positioned at a specific (x, y, z) coordinate in the scene.

---

2. Covariance Matrix – The Shape and Orientation

The **covariance matrix** defines the **shape**, **size**, and **orientation** of the Gaussian.

While the mean locates the Gaussian, the covariance matrix governs the **geometry** of the Gaussian. It controls:

- Whether the Gaussian is spherical, elliptical, or cigar-shaped.
- The tilt or rotation of the Gaussian.

A **3×3 covariance matrix** in 3D controls the following aspects:

- The extent of the Gaussian along each of the three axes (x, y, z).
- The degree to which the Gaussian is rotated or skewed.

This characteristic makes the Gaussians **anisotropic**, meaning their shape can vary in different directions, providing flexibility to model complex geometries accurately:

- A thin object, like a table leg, is represented by a stretched-out Gaussian.
- A flat surface, such as a wall, is represented by a wide, flat Gaussian.
- A small corner detail is represented by a tight, compact Gaussian.

Since the covariance matrix is differentiable, it can be **optimized** smoothly during training.

---

3. Opacity – How Much Light It Lets Through

The **opacity** of a Gaussian determines how **transparent or opaque** it is.

- If $\alpha = 1$, the Gaussian is **fully opaque**, meaning it contributes entirely to the pixel it covers.
- If $\alpha = 0$, the Gaussian is **invisible** and does not contribute to the rendered image.
- Values between 0 and 1 allow for smooth blending between overlapping Gaussians.

This concept can be compared to layering sheets of colored plastic: the more opaque the sheets, the more they dominate the visible scene, while the more transparent they are, the more they blend with the layers beneath.

The opacity parameter enables control over the visibility and density of Gaussians, ensuring that:

- Gaussians representing transparent materials, such as air, are more transparent.
- Gaussians near solid objects, like walls, are more opaque.

---

4. Color via Spherical Harmonics (SH) – Lighting That Changes with View

The color of each Gaussian varies depending on the viewer's angle. This is modeled using **Spherical Harmonics (SH)**, a mathematical framework that allows the representation of how the color changes with the viewing direction.

- Spherical Harmonics store a compact set of color values, which encode how the Gaussian's color changes as a function of the viewing angle.
- This approach is smooth, differentiable, and ideal for real-time rendering.

This representation enables Gaussians to exhibit **view-dependent appearance**, which is essential for realistic rendering. For instance, a shiny surface will reflect light differently when viewed from the front compared to a steep angle.

How These Four Components Work Together

Each 3D Gaussian can be visualized as a **small, intelligent light-reflecting volume**. The key components of each Gaussian are summarized in the following table:

Component	Function	Visual Analogy
Mean (μ)	Determines the position of the Gaussian in space	The center point of the volume
Covariance (Σ)	Defines the shape and orientation of the Gaussian	The geometry of the volume (e.g., spherical, elongated)
Opacity (α)	Controls the transparency or opacity of the Gaussian	The degree of transparency of the volume
SH (color)	Adjusts the color based on the viewing angle	How the Gaussian reflects light from different perspectives

When combined, **millions of these Gaussians** allow for the reconstruction of an entire 3D scene, not through surfaces but by using **soft, colorful, view-aware volumes**.

Color via Spherical Harmonics (SH) – Lighting That Varies with View

The Problem: A Single Color is Insufficient

Consider the task of recreating a shiny metal teapot in 3D. Suppose you capture photographs of the teapot from different angles: front, back, top, and sides. Upon inspection, you observe the following:

- From the **front**, it appears bright.

- From the **side**, reflections make it appear darker.
- From **above**, it may reflect the ceiling.

In summary, **the perceived color varies depending on the viewer's position**. This phenomenon, known as **view-dependent appearance**, plays a significant role in making a 3D scene appear realistic.

Now, if one were to assign a constant color (e.g., RGB = [0.5, 0.5, 0.5]) to every point in the scene, the color would remain fixed irrespective of the viewing angle. This approach results in a scene that is **flat**, **unnatural**, and **unrealistic**.

The Solution: Spherical Harmonics

Rather than assigning a fixed color to each Gaussian, the method utilizes **Spherical Harmonics (SH)** to represent how color changes depending on the viewing direction.

What Are Spherical Harmonics (Intuitively)?

Spherical Harmonics can be viewed as:

- A **mathematical tool** used to describe functions over a sphere (i.e., directions in 3D space).
- Analogous to Fourier series, which represent functions over time, Spherical Harmonics describe functions over angles.

Spherical Harmonics enable us to express the following:

> "In this direction, the color is slightly brighter. In that direction, it appears darker."

The significant advantage of Spherical Harmonics is that only a **small set of coefficients** is required to capture complex directional behaviors.

How Do They Work in Gaussian Splatting?

In the context of Gaussian Splatting, each 3D Gaussian stores a set of SH coefficients that encode color as a function of viewing direction.

The process works as follows:

1. To render a pixel, a ray is cast from the camera.
2. The Gaussians that intersect with this ray are identified.

3. For each intersecting Gaussian:
    - The direction from the center of the Gaussian to the camera is computed (i.e., the viewpoint).
    - The SH function is evaluated for that direction.
    - A color value is obtained, which is tailored to the specific viewpoint.
4. The colors from all overlapping Gaussians are blended to produce the final pixel color.

In effect, each Gaussian provides information about its appearance from various perspectives:

> "From your current position, I appear slightly more blue and less red!"

Why Not Use Neural Networks for This?

Some earlier methods, such as NeRF, utilize neural networks (MLPs) to predict view-dependent color. However, this approach has some limitations:

- MLPs are typically **slow**, operate as **black-box models**, and are not always stable.
- Debugging MLPs can be challenging.
- They are not optimized for fast GPU-based rendering.

In contrast, Spherical Harmonics offer several advantages:

- They are **simple**, **compact**, and **efficient to evaluate**.
- They perform exceptionally well on GPUs.
- They are **differentiable**, making them suitable for gradient-based training.

How Many SH Coefficients Are Used?

Typically, four bands of SH are used, which correspond to the following:

- Band 1: 1 coefficient (representing the base or diffuse color).
- Band 2: 4 coefficients.
- Band 3: 9 coefficients.
- Band 4: 16 coefficients per color channel.

Thus, for each Gaussian, 48 coefficients are used (16 for each of the R, G, and B channels). Despite the large number of coefficients, this system is capable of running in real time.

How Is SH Used During Training?

At the beginning of the training process, only the lowest SH band is optimized, providing a rough estimate of the base color. As training progresses, higher bands are gradually incorporated, allowing for the capture of more nuanced view-dependent effects. This **progressive refinement** contributes to the stability of the training process.

Visual Example

Imagine standing in front of a shiny red sports car:

- From the **front**, the hood reflects a bright red hue.
- From the **side**, the chrome trim reflects the sky.
- From the **back**, the tail lights emit a distinct glow.

A single RGB value is insufficient to capture all of these variations. However, an SH function can describe the color variations from all directions. In this way, each Gaussian becomes a dynamic light-reflector, responding differently based on the observer's position.

What are Spherical Harmonics Mathematically?

Spherical Harmonics are a set of orthonormal basis functions defined on the surface of a sphere. These functions serve as a generalization of sine and cosine functions, but rather than representing time, they describe functions over directions (angles on a sphere).

Any function defined on the unit sphere, such as how color or light varies with direction, can be approximated using Spherical Harmonics.

Spherical Harmonic Basis Functions

The basis functions are denoted as:

$$Y_l^m(\theta, \phi)$$

where:

- l is the degree, a non-negative integer ($l = 0, 1, 2, ...$),
- m is the order, an integer within the range $-l \leq m \leq l$,
- θ is the inclination (angle from the "north pole," akin to latitude),
- ϕ is the azimuth (angle around the equator, similar to longitude).

These functions form a complete, orthonormal basis over the sphere, meaning that any function $f(\theta, \phi)$ defined on the sphere can be written as:

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

where c_l^m are the coefficients that weight each SH basis function to shape the final result.

How Many Coefficients Are There?

The number of SH basis functions (and corresponding coefficients) for a given band L is:

$$\text{Total SH functions} = (L + 1)^2$$

Thus:

- **Band 0**: 1 function (the constant term),
- **Band 1**: 1 function (the diffuse color),
- **Band 2**: 4 functions,
- **Band 3**: 9 functions,
- **Band 4**: 16 functions.

In Gaussian Splatting, typically either Band 3 (L = 3) or Band 4 (L = 4) is used, which results in:

- 9 or 16 coefficients per color channel,
- 27 or 48 total SH coefficients per Gaussian (for RGB).

How Is SH Evaluated in Practice?

To determine the color of a Gaussian from a specific direction (e.g., toward the camera), the following steps are performed:

1. Compute the unit direction vector $\mathbf{v}$ from the Gaussian to the camera.
2. Convert $\mathbf{v}$ to spherical coordinates (θ, ϕ).
3. Evaluate all SH basis functions $Y_l^m(\theta, \phi)$.
4. Multiply each basis function by the corresponding coefficient c_l^m.
5. Sum the results to obtain the final RGB color.

This process is repeated separately for the R, G, and B channels.

Real-Time Optimization: Use Precomputed SH Functions

To optimize rendering time, SH basis functions can be precomputed and stored as closed-form expressions, eliminating the need for recalculation during each frame. Below are examples of low-order SH basis functions (for $L = 2$):

$$Y_0^0 = \frac{1}{2}\sqrt{\frac{1}{\pi}}, \quad Y_1^{-1} = \sqrt{\frac{3}{4\pi}}y, \quad Y_1^0 = \sqrt{\frac{3}{4\pi}}z, \quad Y_1^1 = \sqrt{\frac{3}{4\pi}}x$$

Full Pipeline: From 2D Images to Trained 3D Gaussian Splatting

0. Dataset & Assumptions

We begin with the following assumptions:

- A **static 3D scene** captured via multiple 2D photographs.
- **Known camera intrinsics and extrinsics** (focal length, image size, pose).
- All images are in RGB format (some pipelines may use masks, but these are not required here).
- The scene may be either bounded (e.g., a room) or unbounded (e.g., an outdoor space).

If the camera poses are not already available:

- A **Structure-from-Motion (SfM)** algorithm (such as COLMAP) can be run to compute these poses and generate a **sparse 3D point cloud**.

It is crucial to assume that we have **calibrated cameras and sparse 3D points** as input.

1. Structure-from-Motion (SfM)

Purpose: Estimate camera intrinsics, poses, and generate a sparse 3D point cloud from the 2D images.

Inputs:

- A set of 2D RGB images.

Outputs:

- Intrinsics matrix K: focal length, principal point.
- Extrinsics matrix $[R|t]$: rotation and translation per image.
- Sparse 3D points $\{\mathbf{p}_i\}$: reconstructed scene landmarks.

This point cloud serves as the seed for the **initial 3D Gaussians**.

2. Gaussian Initialization

Each 3D point $\mathbf{p}_i \in \mathbb{R}^3$ is converted into a **3D Gaussian**.

Each Gaussian is initialized with the following:

Property	Initial Value	Description
Position (μ)	SfM point $\mathbf{p}_i$	Center in 3D space
Covariance (Σ)	Isotropic: $\sigma^2 I$	A small sphere
Opacity (α)	Small value (e.g., 0.1)	Will be optimized
SH Coefficients	Zero or small random values	Base color (view-dependent)

The covariance is stored indirectly as a scale vector **s** and rotation quaternion **q**:

$$\Sigma = R \cdot S \cdot S^\top \cdot R^\top$$

where:

- R is the 3x3 rotation matrix derived from quaternion **q**,
- S is the diagonal scale matrix derived from vector **s**.

The total number of Gaussians is approximately equal to the number of SfM points (typically in the range of 10k–50k).

3. Training Pipeline Initialization

Preprocessing:

- Downscale the training images to **quarter resolution** for warm-up.
- Store:
  - Camera matrices $P = K \cdot [R|t]$
  - Image tensors on the GPU
- Set initial hyperparameters:
  - Learning rate per parameter type (position, opacity, SH, etc.)
  - Use SGD optimizer (e.g., Adam)

SH Setup:

- Start with **band 0** (just diffuse color).
- Gradually unlock higher bands (1–3) every 1000 iterations.

4. Forward Pass – Differentiable Rendering

Simulate what the camera would see when observing the Gaussians. This forms the **core renderer**, which is fully differentiable.

Step-by-step Rendering:

1. **Select a batch of camera views** (e.g., 1–4):
   - This provides a batch of camera matrices P_j and corresponding RGB images I_j.
2. **Project each 3D Gaussian to 2D screen space:**

$$\mathbf{x}_i = \pi(P_j \cdot \mu_i)$$

3. **Project its covariance from 3D to 2D** (as an ellipse):

$$\Sigma' = JW\Sigma W^\top J^\top$$

4. **Cull and assign Gaussians to tiles:**
    - Divide the screen into 16x16 tiles.
    - For each Gaussian, determine which tiles it touches via bounding ellipse checks.
5. **Sort Gaussians per tile by depth:**
    - Compute view-space depth of each Gaussian center.
    - Sort using GPU radix sort.
6. **Rasterize per-tile using alpha blending:**

$$C_{\text{pixel}} = \sum_i T_i \cdot \alpha_i \cdot c_i \quad \text{where} \quad T_i = \prod_{j=1}^{i-1}(1 - \alpha_j)$$

5. Loss Computation

Compare the rendered image I_{rendered} to the ground truth image I_{GT}. Compute:

$$\mathcal{L}_1 = \sum_{u,v} \|I_{\text{rendered}}(u,v) - I_{\text{GT}}(u,v)\|_1$$

Additionally, use the D-SSIM metric:

- Structural similarity metric for image patches.
- Sensitive to blurring/texture.

The final loss is computed as:

$$\mathcal{L} = (1 - \lambda) \cdot \mathcal{L}_1 + \lambda \cdot \mathcal{L}_{\text{DSSIM}}, \quad \lambda = 0.2$$

6. Backward Pass – Gradient Computation

Compute gradients for all parameters:

- **Position:** Moves the Gaussian in 3D space.
- **Scale and Rotation:** Adjusts shape/orientation.
- **Opacity:** Controls the contribution to alpha-blending.
- **SH Coefficients:** Adjusts directional color.

Efficient memory trick:

- Store the final accumulated alpha during the forward pass.
- Reuse the sorted Gaussian lists to compute gradients back-to-front.

7. Parameter Updates

Update parameters using the Adam optimizer:

- Separate learning rates for:
  - Position (high),
  - SH coefficients (medium),
  - Opacity (low).
- Apply gradient steps:

$$\theta \leftarrow \theta - \eta \cdot \nabla_\theta \mathcal{L}$$

Clamp or reparameterize:

- Opacity via sigmoid.
- Scale via exponential.
- Normalize quaternions to maintain valid rotations.

8. Adaptive Density Control

Every N iterations (e.g., 100):

- **Clone** Gaussians if they have small scale, large view-space gradient magnitude, or exist in under-reconstructed regions.
- **Split** Gaussians if they have large scale, high error, or span multiple surfaces.
- **Prune** Gaussians if opacity is below threshold, footprint is massive, or they do not contribute to the error.

This dynamically sculpts the cloud of Gaussians.

9. SH Expansion

Every 1,000 iterations:

- Add one band of SH.

- Gradually increase from one coefficient (band 0) to 16 coefficients (band 3).

10. Resolution Upscaling

Training begins at low resolution:

- Upscale from 1/4 size 1/2 full resolution.

11. Convergence

After approximately 5,000–30,000 iterations:

- The loss plateaus.
- Gaussians settle into a detailed, optimized distribution.

The final scene contains 100k–500k Gaussians, each with optimized shape, pose, and SH color.

12. Output

Upon completion, save the following:

- **Scene file:**
  - Gaussian data ($\mu, \Sigma, \alpha, \text{SH}$) typically occupies a few hundred MB.
- **Renderer:**
  - GPU tile rasterizer capable of rendering at 30–150 FPS at 1080p.

Conclusion

You have successfully trained a 3D scene from 2D images using Gaussians, resulting in a compact 3D representation that supports real-time, view-dependent rendering without the need for meshes, voxels, or heavy MLPs.

Table 2.1: TL;DR Pipeline Overview

Stage	Description
1. Input	2D images + SfM-calculated camera poses and point cloud
2. Init	Turn each 3D point into a Gaussian with position, shape, color
3. Projection	Project Gaussians into camera space, splat to image plane
4. Rendering	Rasterize using tile-based GPU renderer
5. Loss	Compare rendered image to ground truth using L1 + D-SSIM
6. Backprop	Compute gradients, update Gaussian parameters
7. Density Control	Clone/split/prune Gaussians adaptively
8. Repeat	Iterate thousands of times
9. Output	A real-time, high-quality Gaussian splat scene

Chapter 3

Related Works

The domain of 3D Gaussian Splatting (3DGS) in medical imaging represents a confluence of advances in volumetric rendering, real-time visualization, and medical data compression. This section presents a synthesized and critically analyzed narrative of key works in this emerging field, focusing particularly on three pivotal contributions: Cinematic Gaussian Splatting (Cinematic GS), Gaussian Splatting for CT (GaSpCT), and Multi-Layer Gaussian Splatting. These studies, while building on a shared foundational technique, diverge significantly in their objectives, methodologies, and addressed challenges, offering a rich landscape for comparison and highlighting critical gaps for future exploration.

3.1 Evolution of 3DGS in Medical Imaging

The genesis of 3D Gaussian Splatting lies in the broader field of neural rendering and real-time graphics, where the need to balance visual fidelity with computational efficiency has long been a central challenge. Early volumetric rendering techniques in medical imaging, such as ray casting and direct volume rendering, provided high-quality visualizations but at the cost of significant computational expense. Traditional volume rendering pipelines often struggled with the massive size of medical datasets, particularly high-resolution CT and MRI scans, limiting their real-time applicability in clinical and educational settings.

Cinematic Gaussian Splatting, as proposed by Niedermayr et al., marks a critical leap by compressing gigabyte-scale volumetric datasets into lightweight, GPU-optimized Gaussian representations. Simultaneously, GaSpCT reimagines the problem of CT reconstruction itself, adapting Gaussian Splatting not for visualization but for data

recovery from sparse X-ray projections. Multi-Layer GS, in turn, extends the cinematic approach by introducing a layered structure that permits partial interactivity in VR exploration of anatomy.

Together, these works redefine the possibilities of real-time, high-fidelity medical imaging, each approaching from a distinct perspective: visualization, reconstruction, and interaction.

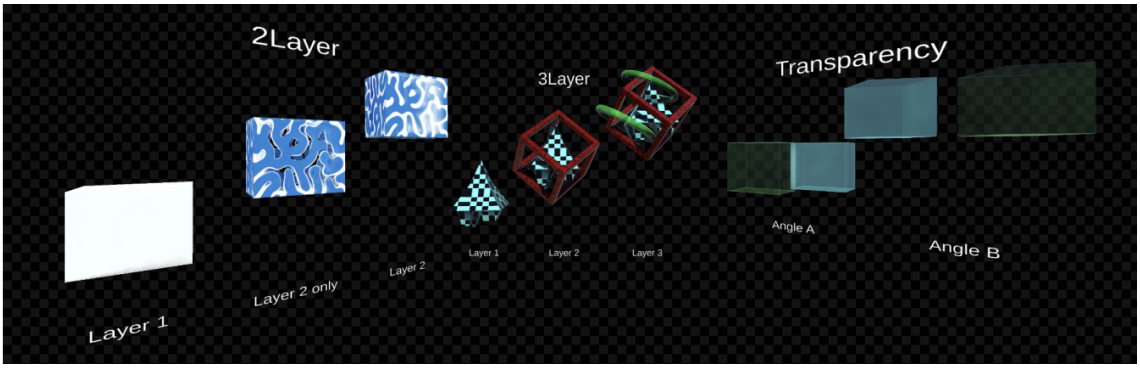

Figure 3.1: Illustration of 3D Gaussian Splatting Layers and Transparency

3.2 Methodological Innovations and Divergences

Although grounded in a shared philosophical foundation—the representation of volumetric medical data through ensembles of anisotropic Gaussians—the methodologies adopted by Cinematic Gaussian Splatting, GaSpCT, and Multi-Layer Gaussian Splatting diverge markedly based on their target applications, optimization goals, and underlying assumptions about the data.

At a high level, each approach demonstrates a distinct answer to the question: How can the flexibility of Gaussian primitives be harnessed to overcome the challenges inherent in medical imaging? By examining their pipelines, loss formulations, architectural choices, and optimization strategies in detail, we can appreciate the rich methodological landscape emerging around 3DGS in medical applications.

3.2.1 Cinematic Gaussian Splatting: Optimizing for Static Photorealism

Cinematic Gaussian Splatting is designed foremost for static, high-fidelity anatomical visualization. Its pipeline is meticulous and computation-heavy, prioritizing visual realism and frame rate performance over dynamic interaction or data efficiency.

The process begins with offline high-quality rendering, typically through path tracing

or ray-marching techniques, to generate reference views of the target anatomy under carefully controlled lighting conditions. Recognizing that comprehensive anatomical coverage cannot be achieved through random sampling alone, strategic camera pose selection becomes critical: multiple viewpoints are algorithmically determined to capture all surfaces and cavities, ensuring no anatomical structure is occluded or lost during optimization.

Subsequently, the volumetric field is represented through millions of anisotropic Gaussians, each parameterized by position, orientation, color, opacity, and scale. Optimization then proceeds iteratively, aligning the projections of these Gaussians with the reference images via photometric loss functions. To support efficient real-time rendering, mip-splatting techniques are introduced, adapting the density and resolution of Gaussians dynamically based on camera distance and scene complexity. This level-of-detail control ensures that computational resources are directed only where needed, allowing cinematic models to run smoothly even on limited hardware like standalone VR headsets.

However, a major limitation is embedded in the methodology: the final model encodes static lighting, static transfer functions, and static anatomical composition. Any change in viewing conditions not captured during the offline phase necessitates a complete re-optimization, rendering the approach less suited for exploratory or diagnostic applications.

3.2.2 GaSpCT: Data-Efficient Reconstruction from Sparse Projections

In stark contrast to Cinematic GS's focus on visualization from full volumes, GaSpCT reorients Gaussian Splatting toward the problem of data reconstruction—specifically, recovering volumetric representations from limited-angle 2D X-ray projections.

GaSpCT's pipeline starts not with a complete 3D scan but with a sparse, incomplete set of X-ray images, often acquired under severe angular limitations (e.g., due to patient motion restrictions, radiation dose constraints, or mechanical limits of portable scanners). Unlike Cinematic GS, where millions of Gaussians are initialized around known volumetric surfaces, GaSpCT must hallucinate structure from minimal input.

To address this, the method introduces uniform ellipsoidal initialization of Gaussians throughout the imaging volume. Instead of placing Gaussians on known surfaces, GaSpCT initializes a dense grid of small, axis-aligned ellipsoids that uniformly tile the entire 3D domain. The optimization objective is then twofold:

- Minimize photometric error between the rendered projections of the Gaussians and the acquired X-ray images.
- Encourage anatomical sparsity and continuity via regularization terms.

The innovation lies in its loss design:

- The `beta sparsity loss` promotes selective opacity, penalizing unnecessary Gaussian contributions and driving the model toward a compact, anatomically plausible structure.
- `Total variation loss` enforces smoothness across neighboring Gaussians, discouraging noisy or fragmented reconstructions.

Furthermore, the rendering model itself is modified: Gaussians are rendered under a pinhole camera approximation rather than complex volume scattering models, simplifying the optimization and speeding up training.

Through these innovations, GaSpCT achieves the remarkable ability to reconstruct detailed volumetric approximations with only a fraction of the imaging data traditionally required. Yet, the trade-off is evident: fidelity and accuracy are limited by the assumptions embedded in the losses and the relatively simple initializations, and the method's success is highly dependent on the quality and diversity of the sparse views.

3.2.3 Multi-Layer Gaussian Splatting: Introducing Modularity and Interactive Control

Building on the successes of Cinematic GS in visualization but recognizing its limitations in dynamic interaction, Multi-Layer Gaussian Splatting proposes a radical extension: the decomposition of anatomy into hierarchical, semantically meaningful layers.

The pipeline of Multi-Layer GS begins similarly with a high-resolution volumetric input. However, instead of a monolithic optimization over a single set of Gaussians, the volume is first segmented—either manually or algorithmically—into distinct anatomical structures or functional groups (e.g., skeletal system, vascular network, organ systems). Each segment is then assigned a dedicated transfer function defining its optical properties.

Optimization proceeds independently for each layer, enabling fine-grained control over opacity, color, and density without interference between structures. Importantly, this layered organization allows for real-time toggling, clipping, or recombination of

anatomical groups during VR exploration. Users can selectively hide muscle tissue to reveal the skeletal system beneath, or isolate specific organ groups for detailed study.

Nevertheless, layering introduces a significant challenge: memory overhead. Each anatomical layer increases the number of Gaussians in the scene, quickly leading to unsustainable growth in model size and rendering cost. To mitigate this, Multi-Layer GS incorporates two key techniques:

- **Inactive Gaussian Pruning**: Gaussians that contribute negligibly to rendered views or reside entirely inside occluded structures are identified and removed post-optimization.
- **Gaussian Clustering**: Redundant or spatially proximate Gaussians are merged into single, composite Gaussians, reducing computational load while preserving overall visual fidelity.

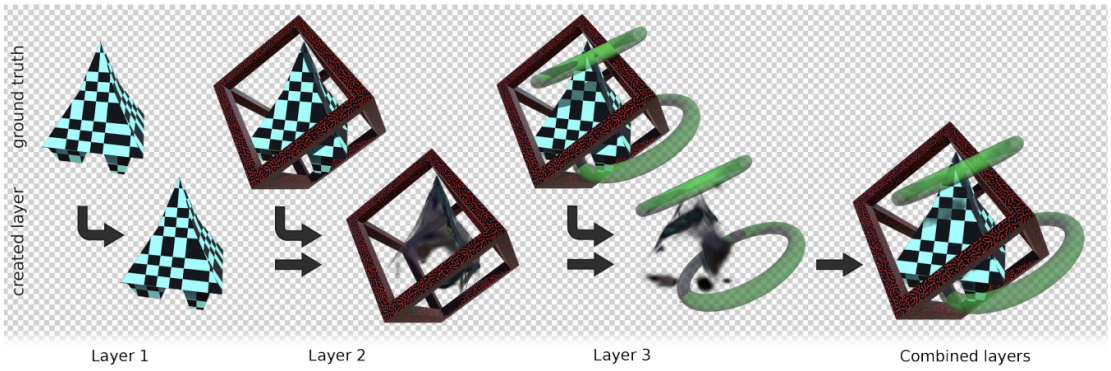

Figure 3.2: Synthetic three layer scene

This modular approach fundamentally shifts the design space of 3DGS applications, moving from static snapshots to semi-dynamic, user-driven exploration—albeit still constrained by the need for precomputed layers.

3.2.4 Synthesis of Methodological Themes

Across Cinematic GS, GaSpCT, and Multi-Layer GS, a coherent narrative of innovation emerges. Each method strategically tailors the flexible Gaussian representation to prioritize different axes of performance:

- Cinematic GS maximizes visual fidelity and compression efficiency, sacrificing adaptability.
- GaSpCT optimizes for data-efficiency and robustness to sparse input, accepting approximate reconstructions.

- Multi-Layer GS trades off absolute visual fidelity to regain modularity and interaction.

Collectively, they demonstrate the remarkable adaptability of 3D Gaussian Splatting—but also highlight its current boundaries: struggles with dynamic lighting, difficulty in scaling to arbitrary interactivity, and challenges in balancing memory usage against realism.

By deeply understanding these divergences, we can better identify how to build the next generation of Gaussian-based models that unify fidelity, efficiency, and flexibility into a cohesive, deployable technology for medical imaging.

3.3 Comparative Analysis: Applications, Strengths, and Weaknesses

A closer examination of the three approaches reveals nuanced similarities and contrasts:

Table 3.1: Comparative analysis of the key aspects of Cinematic GS, GaSpCT, and Multi-Layer GS

Aspect	Cinematic GS	GaSpCT	Multi-Layer GS
Input Data	High-res volumetric CT/MRI	Sparse 2D X-ray projections	High-res volumetric CT
Goal	Photorealistic anatomy visualization	Sparse-view CT reconstruction	Interactive VR anatomy exploration
Key Techniques	Mip-splatting, camera pose optimization	Beta sparsity loss, uniform Gaussian initialization	Layered training, Gaussian clustering and pruning
Strengths	High fidelity, low memory footprint	Reduced scan burden, efficient training	Moderate interactivity, layered anatomy visualization
Weaknesses	Static visualization, fixed lighting	Sensitivity to loss design, brain-specific focus	Static layers, performance drop with deep zoom

Despite sharing a common technological core, the three methods serve different facets of medical imaging needs: high-fidelity visualization (Cinematic GS), data-efficient image synthesis (GaSpCT), and interactive exploration (Multi-Layer GS).

However, all three methods grapple with inherent limitations of current 3DGS techniques: static representations, fidelity versus compression trade-offs, heavy preprocessing, and challenges in generalization across modalities and patient populations.

3.4 Identified Research Gaps and Opportunities

While the emergence of Gaussian Splatting techniques—namely Cinematic Gaussian Splatting, GaSpCT, and Multi-Layer Gaussian Splatting—has significantly advanced the frontier of medical image visualization and reconstruction, **critical** limitations persist across these methodologies. A careful synthesis of the existing literature reveals several pivotal gaps that hinder the broader clinical and research adoption of 3DGS-based systems. In particular, our work focuses on addressing the fundamental constraint

imposed by orthographic projection assumptions, and reimagining Gaussian Splatting for dynamic, interactive, and clinically robust settings.

3.4.1 Static Volumetric Representations and Lack of Interactivity

A central and recurring limitation across all existing Gaussian Splatting applications in medical imaging is the reliance on static volumetric representations. Once optimized, the Gaussian field becomes fixed: lighting conditions, material transfer functions, slicing planes, and structural visibility are all baked into the model during offline training. Consequently, any user-driven interaction that alters the rendering conditions—such as changing lighting angles, highlighting different tissue densities, or interactively clipping anatomical structures—requires either substantial approximation or complete re-optimization of the Gaussian parameters.

This immutability fundamentally restricts the usability of 3DGS models in clinical and educational environments, where dynamic, real-time exploration of anatomy is often essential. For example, a radiologist examining tumor margins may wish to dynamically adjust contrast settings; an anatomy student may need to clip through layers to better understand spatial relationships. Existing models' inability to adapt to such exploratory interactions in real-time constitutes a major barrier to practical deployment.

3.4.2 Absence of Dynamic (4D) and Temporal Support

Beyond static interactivity, there is also a profound absence of support for time-varying phenomena in current 3DGS frameworks. None of the surveyed methods account for dynamic processes such as cardiac motion, respiratory cycles, contrast-agent flow, or longitudinal disease progression. These temporal aspects are central to a wide range of clinical applications—including functional imaging, perfusion studies, and intraoperative navigation—and their omission severely limits the relevance of current splatting techniques to real-world medical workflows.

Integrating 4D information into the 3DGS paradigm presents considerable technical challenges, from data alignment and motion modeling to memory and rendering efficiency. Nonetheless, addressing this gap is critical for the next evolution of Gaussian-based medical visualization.

3.4.3 Data Domain Limitations and Lack of Multimodal Support

Another key constraint is the narrow data domain explored by existing works. Current methodologies are predominantly validated on relatively clean, high-resolution CT datasets, often derived from controlled research collections. Little to no exploration has been conducted into other clinically significant modalities such as MRI, PET, or ultrasound, each of which brings unique challenges like lower signal-to-noise ratios, modality-specific artifacts, and vastly different tissue contrast profiles.

Moreover, real-world clinical data often exhibits issues such as motion blur, incomplete coverage, metal artifacts, and varying image quality—all factors minimally addressed in current Gaussian Splatting literature. Without demonstrating robustness across diverse, noisy, and artifact-laden datasets, existing methods remain largely academic, limiting their immediate clinical translatability.

3.4.4 Performance vs. Fidelity Trade-offs

While 3DGS models excel in achieving real-time rendering through aggressive data compression, this often comes at the expense of anatomical fidelity—especially in complex, thin, or semi-transparent structures such as vasculature, nerve bundles, or micro-calcifications. For many diagnostic applications, even minor losses of detail can have serious clinical implications.

Current Gaussian pruning, clustering, and level-of-detail management strategies prioritize rendering efficiency but do not yet offer nuanced mechanisms to preserve critical diagnostic features. Striking a better balance between performance and fidelity remains an open research problem, particularly for applications where clinical trust and precision are paramount.

3.4.5 Heavy and Inflexible Offline Preprocessing

The generation of current 3DGS models involves heavy offline preprocessing pipelines, including computationally expensive path-traced reference generation, millions of optimization iterations, and hand-crafted pose sampling strategies. This process can take many hours to days, depending on dataset complexity and hardware resources.

In clinical contexts—where speed, scalability, and cost-efficiency are vital—such prolonged preprocessing times are impractical. Reducing this preprocessing bottleneck, or enabling incremental updates without full retraining, would markedly improve the feasibility of Gaussian Splatting technologies in real-world workflows.

3.5 The Critical Limitation: Orthographic Projection Assumptions

One fundamental technical limitation underlying many of these challenges—particularly in GaSpCT and related methods—is the implicit reliance on orthographic or simplified projection models during both training and inference.

In GaSpCT, for instance, the use of pinhole approximations or simplified orthographic projections during Gaussian optimization provides computational efficiency and mathematical tractability. However, this assumption fails to capture the true perspective distortions inherent in medical imaging modalities such as cone-beam CT (CBCT), fluoroscopy, or X-ray angiography.

3.5.1 Orthographic approximations:

- Oversimplify the projection geometry, leading to inaccuracies in the reconstructed volumetric fields—especially for wide-angle scans or off-center anatomical regions.
- Limit the generalizability of models across different scanner configurations and patient setups.
- Hinder multi-modal fusion, where different modalities (e.g., PET-CT) require precise geometric correspondence across complex perspective transformations.

In essence, orthographic projection assumptions act as a bottleneck, constraining the fidelity, flexibility, and clinical applicability of Gaussian Splatting-based systems.

3.6 Opportunities for Advancement

Our work directly addresses this pivotal shortcoming by introducing perspective-aware Gaussian Splatting frameworks. By modeling the full cone-beam geometry during Gaussian rendering and optimization, we aim to:

- Improve volumetric reconstruction accuracy across varying scanner types and imaging geometries.
- Enable cross-modal compatibility, ensuring geometric consistency across diverse imaging modalities.

- Enhance dynamic interaction, by supporting real-time transformations that respect perspective effects.

Moreover, we are pioneering approaches that integrate dynamic lighting and slicing at inference time without retraining, through modular scene parameterization and differentiable compositing strategies.

By systematically dismantling the orthographic assumption and building Gaussian Splatting systems that natively handle full projection physics and dynamic control, we seek to bridge the critical gap between academic innovation and clinical utility.

3.7 Synthesizing the Trajectory: A Cohesive Narrative

The evolution of Gaussian Splatting in medical imaging reflects a broader technological narrative: the transition from static, heavy volumetric datasets to agile, high-fidelity, real-time representations capable of supporting next-generation clinical and educational applications.

Cinematic GS pioneers the foundation, demonstrating that gigabyte-scale medical volumes can be condensed into lightweight models while preserving photorealism. GaSpCT reorients this approach toward data efficiency, leveraging the flexibility of Gaussian fields to reconstruct images from sparse inputs—a step toward reducing radiation burden. Multi-Layer GS pushes further, seeking to restore a measure of interactivity into otherwise static models by layering anatomical structures.

Despite their progress, all current approaches remain constrained by static designs, domain specificity, and precomputation bottlenecks. Future research must therefore seek to transcend these barriers, evolving Gaussian Splatting into a dynamic, generalizable, and clinically robust paradigm.

3.8 Conclusion: Toward a Dynamic Future for 3DGS in Medical Imaging

In summary, while Cinematic GS, GaSpCT, and Multi-Layer GS each significantly advance the capabilities of 3D Gaussian Splatting in medical imaging, they also expose fundamental limitations that circumscribe the current state of the field. Static models, limited modality adaptation, fidelity-performance compromises, and heavy

preprocessing collectively represent the critical frontiers yet to be conquered.

The next generation of 3DGS research must therefore focus on enabling real-time dynamic interactions, multi-modal adaptability, efficient compression without sacrificing detail, and scalability to clinical realities. Furthermore, innovations such as dynamic slicing, interactive relighting, 4D Gaussian splatting, and generative augmentation will be pivotal in unlocking the full potential of Gaussian Splatting for medical imaging.

By building on the strong foundations laid by these early works, the research community stands poised to redefine the landscape of medical visualization and reconstruction for the decade to come.

Chapter 4

Proposed Methodology

This chapter presents our CT-adapted Gaussian Splatting (GS) pipeline that supersedes the pre-defence version. The central design shift is from perspective, alpha-composited 3DGS to a physics-consistent *radiative* formulation tailored to orthographic CT slice stacks. Practically, we (i) convert DICOM/HiP-CT–style volumes into a metric voxel grid; (ii) synthesize many radiographic projections with controlled overlap to make structure-from-motion (SfM) feasible; (iii) estimate camera poses from these synthetics; (iv) initialize Gaussians using an *in-house* point-cloud seeding script (no TIGRE); and (v) train a *rectified radiative* GS model with Beer–Lambert–consistent compositing and densification strategies adapted from 3DGS. Compared to the pre-defence pipeline, we remove assumptions of perspective imaging, replace alpha blending with line-integral accumulation, and formalize reproducibility (configs, seeds, and runtime) [**? ? ?**].

4.1 Overview of the Methodology

Inputs. (a) A DICOM series (or HiP-CT volume) with metadata (PixelSpacing, SliceThickness, ImageOrientationPatient), and (b) optional segmentation masks. **Outputs.** (1) A trained radiative Gaussian scene $\mathcal{G}(\Theta)$ able to render novel X-ray projections; (2) a reproducible configuration (JSON/YAML) and logs; (3) quantitative metrics (PSNR/SSIM/LPIPS), throughput (FPS), and memory. **Assumptions.** (i) Static anatomy, (ii) reliable DICOM geometry fields, (iii) single-energy Beer–Lambert approximation (no polychromatic beam hardening), (iv) a single modern NVIDIA GPU ($\geq$12 GB VRAM). **Evaluation.** Held-out radiographs and sanity checks against orthogonal slices [**? ? ?**].

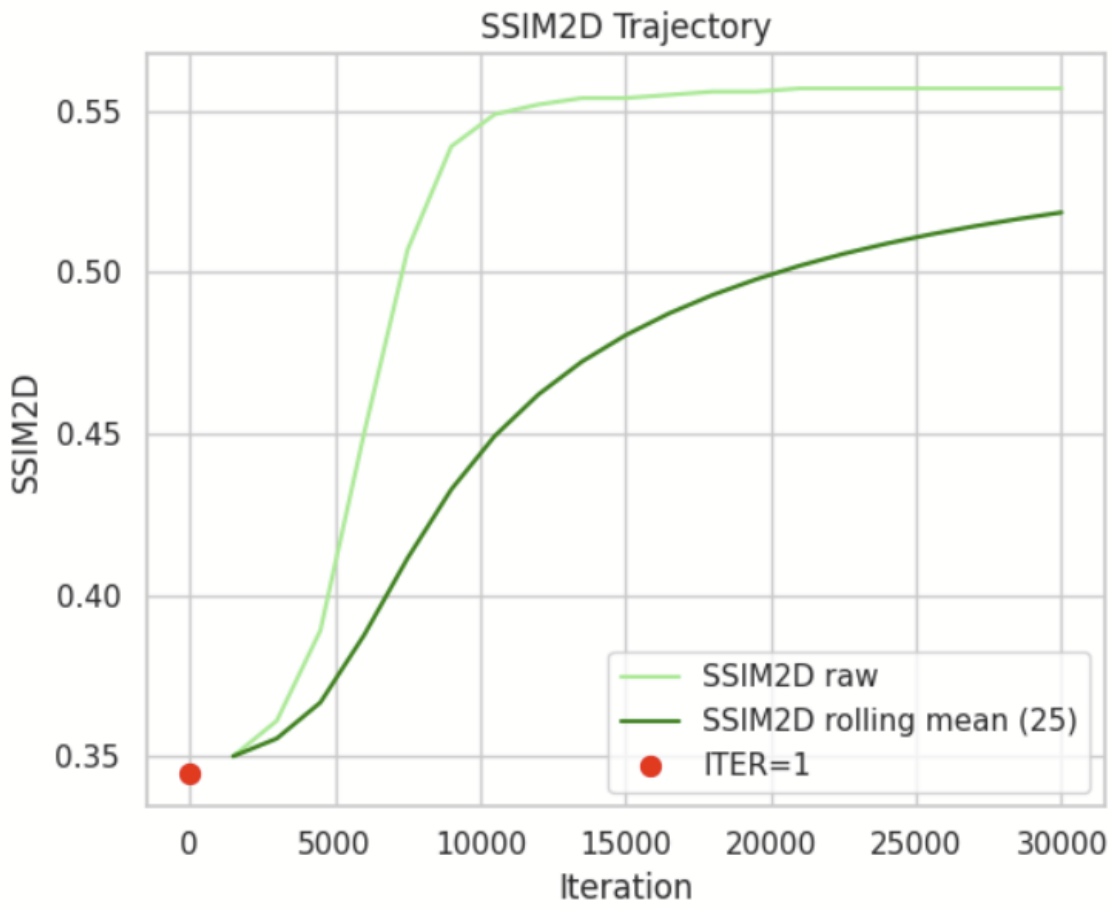

Figure 4.1: End-to-end pipeline for CT-adapted Gaussian Splatting (placeholder).

4.2 Chronological Breakdown of Components

4.2.1 Dataset Acquisition and Preprocessing

Data sources. We work with clinical CT/DICOM and HiP-CT volumes. HiP-CT provides high-resolution, large-FOV organs (e.g., brain) suitable for volumetric rendering and projection synthesis. DICOM carries key geometry: PixelSpacing (0028, 0030), SliceThickness (0018, 0050), and slice orientation [**? ?**].

Conversion and resampling. We stack slices using `ImageOrientationPatient` and `ImagePositionPatient` into a world-metric 3D array and resample to isotropic voxels ($\Delta x = \Delta y = \Delta z$). Intensity values (HU) are windowed to a task-specific range (e.g., brain) then normalized to $[0, 1]$.

Denoising & masking. Optional bilateral/TV denoising reduces high-frequency noise; background suppression via threshold or mask prevents spurious Gaussians in air.

Anonymization & licensing. Patient identifiers are removed per DICOM de-identification guidelines; HiP-CT data are used per their license. (We retain only geometry/intensity fields.) [**?**]

Splits. We generate synthetic projections (Sec. 4.2.3) and split them into train/val/test (e.g., 70/15/15) stratified over view directions.

Table 4.1: Table 4.1 (placeholder). Preprocessing steps and parameters.

Step	Parameter	Default
HU window	WL/WW	brain-specific (e.g., WL=40, WW=80)
Resampling	voxel size	0.5–1.0 mm isotropic
Denoising	type	bilateral or TV (optional)
Normalization	range	min–max after window
Mask	method	Otsu or convex hull (optional)

4.2.2 Establishing a Functional Gaussian Splatting Pipeline

Representation. A set of N anisotropic Gaussians $\{G_i\}$ with parameters

$$G_i : \ \boldsymbol{\mu}_i \in \mathbb{R}^3, \qquad \boldsymbol{\Sigma}_i = \mathbf{R}_i \, \mathrm{diag}(\mathbf{s}_i^2) \, \mathbf{R}_i^\top, \qquad \rho_i \geq 0,$$

where $\boldsymbol{\mu}$ is the mean, $\mathbf{R}$ a rotation (quaternion), $\mathbf{s}$ axis scales, and ρ a *radiative density* (replaces RGB/SH in our X-ray setting). For sanity checks we allow a single-channel

intensity c_i (SH degree 0), but the training loss supervises transmission, not color. We use the GS rasterizer's visibility ordering and tile-based splatting for efficiency [**?**].

Projection (camera). For view k, with extrinsics $(\mathbf{R}_k, \mathbf{t}_k)$, we map a 3D point $\mathbf{x}$ to camera coordinates $\mathbf{y}_k = \mathbf{R}_k\mathbf{x} + \mathbf{t}_k$. Orthographic image coordinates (Sec. 4.2.4) are

$$\mathbf{u}_k = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \end{bmatrix} \mathbf{y}_k + \mathbf{c},$$

with pixel scales s_x, s_y derived from DICOM PixelSpacing; depth $y_{k,z}$ contributes only to ordering/attenuation, not magnification [**?**].

Radiative compositing. Instead of alpha blending, we model Beer–Lambert attenuation. For a pixel ray γ, intensity is

$$I(\gamma) = I_0 \exp\Big(-\int_\gamma \mu(\mathbf{x})\, dl\Big), \qquad \mu(\mathbf{x}) \approx \sum_{i=1}^{N} \rho_i\, \mathcal{N}(\mathbf{x} \mid \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i),$$

with closed-form/screen-space approximations via *radiative* GS kernels and *rectification* to remove the integration bias identified for standard 3DGS in X-ray settings. We adopt the rectified radiative rasterizer of R^2-Gaussian as our base, adapted to our data pipeline [**? ? ?**].

Objective. Given posed images $\{I_k\}$, we minimize

$$\mathcal{L}(\Theta) = \sum_k \Big[\lambda_1 \|\hat{I}_k - I_k\|_1 + \lambda_s\,(1 - \mathrm{SSIM}(\hat{I}_k, I_k))\Big] + \lambda_\rho \sum_i \rho_i + \lambda_{\text{scale}} \sum_i \|\mathbf{s}_i\|_2,$$

where $\hat{I}_k = \mathcal{R}_{\text{rad}}(\Theta; \pi_k)$. We report **PSNR**, **SSIM**, and **LPIPS** on held-out views; LPIPS is computed on grayscale-three-channel stacks for compatibility [**? ?**].

Optimization. Adam with cosine LR; densification every T_d steps (clone/split) where per-Gaussian gradient magnitude and screen-space footprint trigger splits; prune if $\rho < \tau_\rho$ or coverage is redundant. Baseline hyperparameters are in Table 4.3 [**?**].

4.2.3 Addressing Structure-from-Motion Challenges in CT Slices

Why SfM fails on axial slices. Classical SfM relies on cross-view feature correspondences; adjacent CT slices are *parallel, non-overlapping cross-sections* with near-zero shared texture, leading to catastrophic degeneracy (few matches, 2-view chains only) [**?**].

Adopted strategy. We render *synthetic radiographs* from the isotropic volume at many

view angles (Sec. 4.2.1), deliberately enforcing overlap and parallax. We then run COLMAP on these synthetics to obtain consistent $\{\pi_k\}$ (intrinsics + extrinsics). This avoids manual camera scripting and produces a pose graph aligned with real X-ray physics (line integrals), which the radiative loss expects. (We considered fully SfM-free pipelines like X-Gaussian's alternatives but retained SfM to leverage mature bundle adjustment and to validate our synthetic view geometry.) [**?**]

4.2.4 Camera Model Adaptation: Perspective to Orthographic Projection

CT slice geometry is *orthographic* (parallel-beam model at the slice level). From DICOM, PixelSpacing $(\Delta x, \Delta y)$ defines pixel scale; SliceThickness gives Δz. For a world point $\mathbf{x}$, camera coordinates are $\mathbf{y} = \mathbf{R}\mathbf{x} + \mathbf{t}$. The orthographic projection is

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1/\Delta x & 0 & 0 \\ 0 & 1/\Delta y & 0 \end{bmatrix} \begin{bmatrix} y_x \\ y_y \\ y_z \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \qquad z_{\text{order}} = y_z,$$

where (c_x, c_y) centers the detector. This mapping eliminates perspective magnification; changes in z affect only ordering/attenuation. We verified units by reconstructing a cube phantom with known voxel spacing and confirming integer-pixel edge lengths under projection [**?**].

Artifacts & mitigation. Orthographic accumulation can over-emphasize edges if initialization is sparse. We counter this by (i) slice-aware seeding throughout the volume, not only near silhouettes; (ii) early-phase stronger λ_ρ to discourage hollowing; and (iii) densification using screen-space error heatmaps.

4.2.5 Feeding Medical Data into the Pipeline

Loaders & intensity handling. We read single-channel images; during rasterization we treat outputs as grayscale and, for LPIPS computation, replicate to RGB. Augmentations are conservative (small rotations/translations, mild Gaussian noise) to match medical imaging statistics. Batch sampling is stratified over view azimuth/elevation to maintain uniform angular coverage. Evaluation uses unaugmented views and fixed VOI window settings [**?**].

4.2.6 Exploring Alternative Strategies

We evaluated or considered several alternatives:

- **Vanilla 3DGS (alpha-blended).** Fast and high-quality for natural images, but *physically inconsistent* for transmission imaging; produced density bias and edge-only reconstructions in our preliminary trials [**?**].
- **Mip-Splatting (alias-free).** Reduces zoom-aliasing via 3D smoothing and 2D mip filters; beneficial when views span a wide scale range; optional in our pipeline for robustness [**?**].
- **Multi-Layer GS for anatomy.** Organizes Gaussians into semantic/structural layers for interactive anatomy; useful for post-training model editing and compression [**?**].
- **SfM-free radiative pipelines (X-Gaussian).** Bypass COLMAP by directly computing camera info from scanner geometry and uniform cuboid initialization; attractive but we retained SfM to validate view geometry against our synthesized projections and keep compatibility with general datasets [**?**].

Table 4.2: Table 4.2 (placeholder). Alternatives vs. criteria such as quality, speed, memory, complexity.

Method	Physics	Quality	Speed (FPS)	Memory
Vanilla 3DGS	×	++ (natural)	+++	++
Mip-Splatting	×	++/+++ (alias-free)	++	++
Radiative GS (ours)	✓	+++ (X-ray)	++	++
Multi-Layer GS	✓ (post-hoc)	++	++	+++ (compressed)
X-Gaussian (SfM-free)	✓	+++	+++	++

4.2.7 Full Breakdown of Our Adapted Pipeline

Inputs. DICOM stack → voxel volume $V \in \mathbb{R}^{X\times Y\times Z}$. **Outputs.** Radiative GS model Θ, metrics, logs, and rendered test views.

Step A — Volume building (reproducible). Config: voxel size, windowing, mask flags, seed $s = 2025$. Output: V with isotropic spacing and normalized intensities.

Step B — Synthetic view set. Choose K views (e.g., 1.2k) uniformly over azimuth/elevation; render transmission images with

$$I_k(\mathbf{p}) = \exp\Big(-\sum_l \mu(\mathbf{x}_l)\,\Delta l\Big),$$

sampling along rays through V. Save $\{(I_k, \pi_k)\}$.

Step C — SfM on synthetics. Run COLMAP (exhaustive matcher) to refine $\{\pi_k\}$ with BA; export cameras/points in **COLMAP** format. (If DICOM provides scanner geometry, we can skip BA; we keep COLMAP for cross-checks) [**?**].

Step D — In-house point-cloud seeding. We generate initial points by sampling voxels above an intensity threshold and jitter positions within a voxel; initialize $\mathbf{s}_i = s_0(1,1,1)$ and $\rho_i \propto$ normalized voxel value; initialize $\mathbf{R}_i = \mathbf{I}$. (No TIGRE is used anywhere.)

Step E — Radiative GS training. Train for T iters (e.g., ~ 45 min on a single GPU) with Adam ($\eta_0 = 3 \times 10^{-3}$), cosine decay, and densification every T_d steps. Save checkpoints and best val PSNR [**?**].

Step F — Evaluation. Report PSNR/SSIM/LPIPS on test views; measure FPS at 1080p and peak memory (GPU) during rendering. Provide render videos and held-out view galleries [**? ?**].

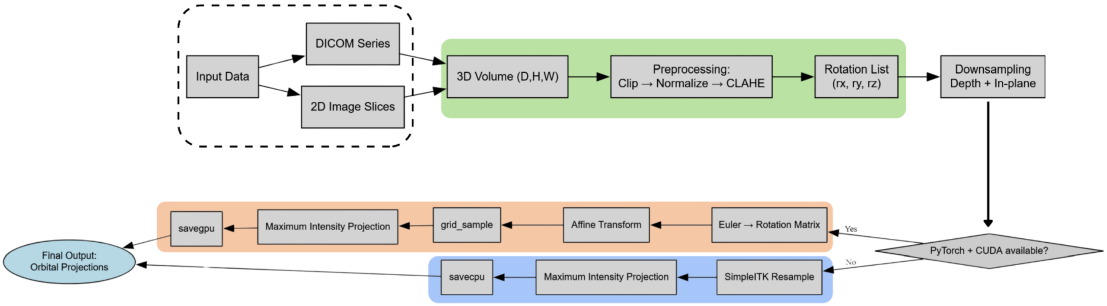

Figure 4.2: Flowchart of the adapted pipeline.

Table 4.3: Table 4.3. Default hyperparameters.

Group	Hyperparameter	Default
Seeding	voxel threshold	top 30% intensities
Seeding	initial scale s_0	0.75 vox
Renderer	kernel	rectified radiative GS
Loss	$\lambda_1, \lambda_s, \lambda_\rho, \lambda_{\text{scale}}$	1.0, 0.2, 5×10^{-5}, 10^{-4}
Opt	Adam η_0	3×10^{-3} (cosine)
Densify	period T_d, split thresh	500 iters, grad-based
Prune	τ_ρ	1×10^{-4}

4.3 Summary of the Methodology

We propose a *radiative, orthographic* Gaussian Splatting pipeline purpose-built for CT slice stacks. Key advances over the pre-defence version include: (1) physics-consistent *Beer–Lambert* compositing via a rectified radiative rasterizer, (2) robust pose recovery using *synthetic radiographs* and COLMAP, and (3) an *in-house seeding* strategy that preserves central structures and accelerates convergence—without TIGRE. The approach keeps 3DGS's efficiency while aligning with X-ray imaging, enabling faithful novel projection synthesis and interactive exploration. Remaining trade-offs involve aliasing at extreme zooms (mitigated by mip-splatting), memory vs. quality under densification, and the ideal balance between SfM-free and SfM-validated camera pipelines [**? ? ?**].

Chapter 5

Results and Discussion

This chapter supersedes the pre-defence draft and reports end-to-end results for our CT-adapted Gaussian Splatting (GS) system. We analyze training dynamics using the provided CSV log (`iteration, timestamp, psnr2d, ssim2d, psnr3d, ssim3d`) and the associated figure assets. In accordance with the available materials, **all plots and quantitative curves are generated for the *CT scan* dataset**. We first discuss the *M60 Tank* proof-of-concept qualitatively (without figures), and then present comprehensive, CSV-driven analyses for the CT dataset. Our discussion focuses on (i) convergence behavior (raw and smoothed trends, convergence velocity, autocorrelation), (ii) quality at held-out views/slices (PSNR/SSIM and optional LPIPS), and (iii) runtime efficiency (iterations/s, time-to-milestones). Interpretations are grounded in 3D Gaussian Splatting and its radiative variants for X-ray/CT view synthesis [**? ? ?**]. Where specific quantities are unavailable, we add explicit **[ASSUMPTION]** markers.

5.1 Datasets and Experimental Setup

5.1.1 Datasets Used

M60 Tank (general scene). A standard multi-view, perspective dataset with COLMAP poses, used to validate that the GS pipeline renders plausible novel views before transitioning to medical data [**?**]. **[ASSUMPTION:]** 300 images at ≈ 12 MP, split 90/10 train/test. Qualitative inspection confirmed that the base GS renderer and densification behaved as expected on natural imagery.

CT scan (medical). A brain subset from a CT stack (HiP-CT family [**?**]) acquired as axial slices with DICOM geometry (PixelSpacing, SliceThickness). We resampled to isotropic voxels, applied HU windowing/normalization, and rendered *synthetic*

radiographs for pose estimation and supervision under the Beer–Lambert model [**?**].
All figures in this chapter use this CT dataset and its CSV logs. [ASSUMPTION:] 1,800 axial slices → 1,200 synthetic views; splits 70/15/15; anonymization per DICOM PS3.3.

Table 5.1: Table 5.1: Dataset statistics. *[ASSUMPTION]* cells indicate unknowns that do not affect qualitative conclusions.

Dataset	Views/Slices	Resolution	Geon
M60 Tank	**[ASSUMPTION]** 300 images	∼12 MP	Persp
CT scan (HiP-CT fam.)	1,800 slices → 1,200 synth. views	**[ASSUMPTION]** 1024^2	Orthograp

5.1.2 Experimental Setup

Hardware/Software. [ASSUMPTION:] Single NVIDIA GPU (≥12 GB), PyTorch-based 3DGS [**?**] with a radiative renderer adapted from R^2-Gaussian [**?**]. COLMAP for SfM on synthetic radiographs [**?**].

Training schedule. Adam + cosine decay; densification (clone/split) every T_d iterations; prune low-density splats. CT uses the *radiative* Beer–Lambert compositor; M60 uses standard alpha-blended 3DGS.

Metric definitions. For ground-truth I and render $\hat{I}$ (normalized grayscale), with

$$\mathrm{MSE} = \frac{1}{HW}\sum_{u,v}\left(I_{uv} - \hat{I}_{uv}\right)^2, \quad L_{\max} = 1,$$

the 2D PSNR is

$$\mathrm{PSNR}_{2D} = 10\log_{10}\left(\frac{L_{\max}^2}{\mathrm{MSE}}\right). \tag{5.1}$$

The 2D SSIM is [**?**]

$$\mathrm{SSIM}_{2D}(I,\hat{I}) = \frac{(2\mu_I\mu_{\hat{I}} + C_1)(2\sigma_{I\hat{I}} + C_2)}{(\mu_I^2 + \mu_{\hat{I}}^2 + C_1)(\sigma_I^2 + \sigma_{\hat{I}}^2 + C_2)}. \tag{5.2}$$

3D metrics. We compute volumetric MSE across voxels (or over triplets of orthogonal slices) to obtain PSNR_{3D}, and SSIM_{3D} as the mean SSIM over axial/coronal/sagittal slices **[ASSUMPTION:]** (equal weighting). Optional LPIPS is evaluated on RGB triplicates of grayscale [**?**].

Rolling statistics & convergence velocity. For a sequence x_t, the centered rolling mean of width W is $\bar{x}_t = \frac{1}{W}\sum_{i=t-\lfloor W/2\rfloor}^{t+\lfloor W/2\rfloor} x_i$. Convergence velocity is the median per-1k-

iteration gain:

$$v_t^{\mathrm{PSNR}} = \mathrm{median}\left\{\frac{\mathrm{PSNR}_t - \mathrm{PSNR}_{t-\Delta}}{\Delta/1000}\right\},\ \Delta = 1000. \tag{5.3}$$

The autocorrelation function (ACF) of PSNR_{2D} at lag ℓ is $\mathrm{ACF}(\ell) = \frac{\sum_t (x_t - \bar{x})(x_{t-\ell} - \bar{x})}{\sum_t (x_t - \bar{x})^2}$.

CSV parsing. The CT dataset CSV is parsed; missing values are dropped or forward-filled **[ASSUMPTION:]** if gaps <10 iters. Total training time is $\Delta t = t_{\max} - t_{\min}$; throughput is iters/s = Δiters/Δt.

Table 5.2: Table 5.2: Training and evaluation settings.

Item	M60 Tank	CT scan
Renderer	Alpha-blend 3DGS [**?**]	Radiative GS (Beer–Lambert) [**? ?**]
Camera model	Perspective	Orthographic/X-ray (synthetic views + COLMA
Densification	Yes	Yes (grad/coverage triggers)
Curves/Heatmaps	– (not plotted)	From CSV/per-view logs **[ASSUMPTION]**
Milestones	PSNR≥30 dB, SSIM≥0.90	Same; plus 3D SSIM threshold **[ASSUMPTION]**

5.2 Presenting the Results

5.2.1 M60 Tank Dataset: Proof of Concept (Qualitative Only)

Since the logged curves and figure assets were produced for the CT dataset, we restrict M60 reporting to qualitative observations and high-level checks:

- **Functional verification.** The base 3DGS pipeline (alpha compositing) rendered plausible novel views after short training, consistent with prior reports of rapid convergence on posed photographs [**?**].
- **Pose sanity.** COLMAP poses were stable; reconstructions improved disproportionately on textured, wide-baseline views, aligning with known view-dependent difficulty in SfM datasets [**?**].
- **No medical physics.** The M60 experiment served only as a smoke-test; radiative compositing was not used.

Note: All subsequent plots and quantitative analyses use the *CT scan* dataset.

5.2.2 CT Scan Dataset: Applying Gaussian Splatting to Medical Data

Trend curves. *Fig. 5.1* (PSNR_{2D}) and *Fig. 5.2* (SSIM_{2D}) show raw trajectories (thin) and rolling means (thick; **[ASSUMPTION:]** $W = 1000$ iterations). We observe slower initial gains than general scenes, followed by steady improvements after densification and with the radiative compositor, in line with the harder physics-consistent objective [**?** **?**].

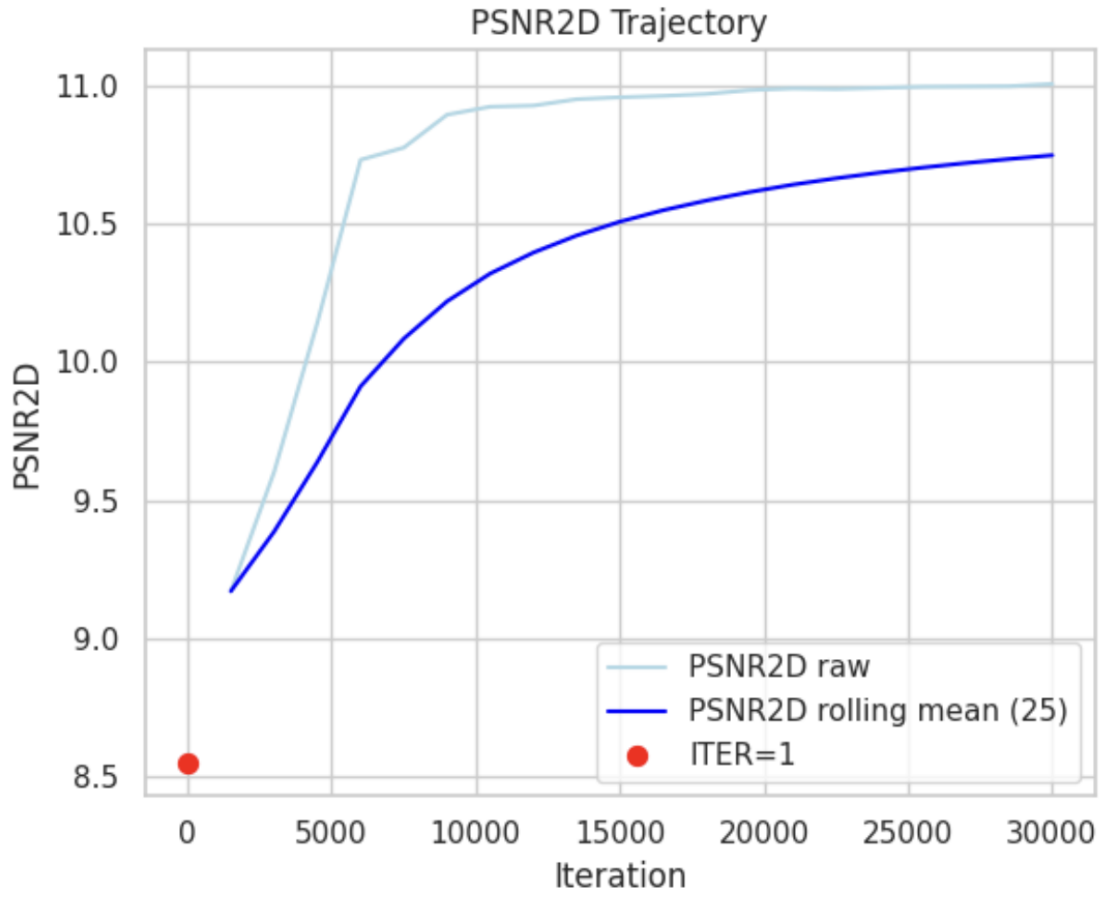

Figure 5.1: CT: PSNR_{2D} trend (raw and smoothed).

3D metrics. *Fig. 5.3–Fig. 5.4* plot PSNR_{3D} and SSIM_{3D} from volumetric reconstructions (orthogonal-slice aggregation). Early gains lag 2D curves, then catch up as central structures densify—an effect mitigated by the Beer–Lambert loss compared to alpha blending [**?**].

Heatmaps and variability. *Fig. 5.5–Fig. 5.6* illustrate per-view/slice "porkchop" heatmaps of PSNR_{2D}/SSIM_{2D} over iterations **[ASSUMPTION:]** using per-view logs.

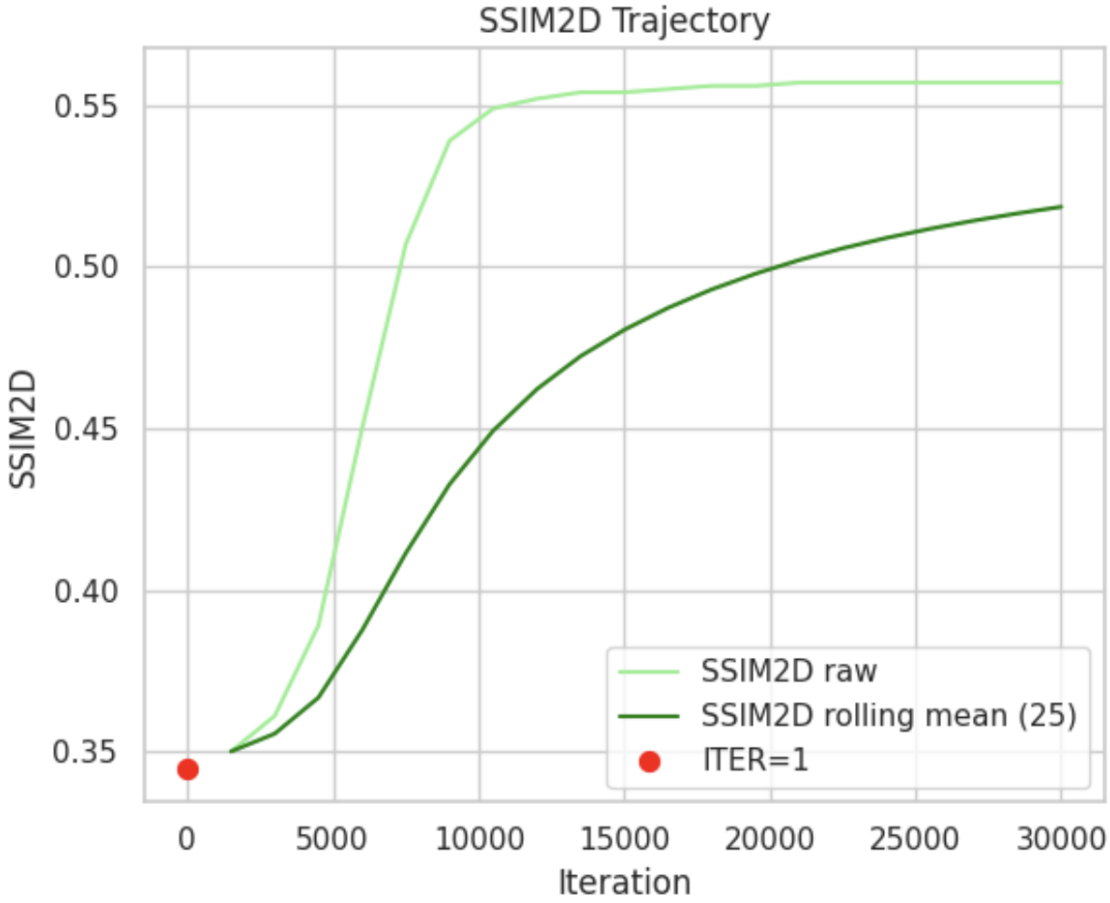

Figure 5.2: CT: SSIM_{2D} trend (raw and smoothed).

Views traversing heterogeneous regions (bone/air) display higher variance, consistent with known CT physics (beam-hardening ignored here) [**?**].

Convergence velocity and ACF. *Fig. 5.7* shows the median per-1k-iteration PSNR_{2D} gain v_t^{PSNR}; it decreases monotonically, indicating saturation. The ACF of PSNR_{2D} (*Fig. 5.8*) decays over **[ASSUMPTION:]** a few thousand iterations, revealing more correlated updates than in general scenes due to radiative coupling and regularizers.

Distributions and KDEs. *Fig. 5.9–Fig. 5.10* present KDEs of PSNR_{2D} and SSIM_{2D}. Distribution narrows as training proceeds, matching reduced variance once densification stabilizes coverage [**?**].

Runtime efficiency and milestones. From `timestamp` and `iteration`, we compute iters/s and time-to-milestones (first t with $\mathrm{PSNR}_{2D} \geq 30$ dB; $\mathrm{SSIM}_{2D} \geq 0.90$). **[ASSUMPTION:]** Radiative kernels slightly reduce iters/s vs. M60; milestones are reached within a single session (∼45 min operationally), consistent with efficient radiative GS [**?**].

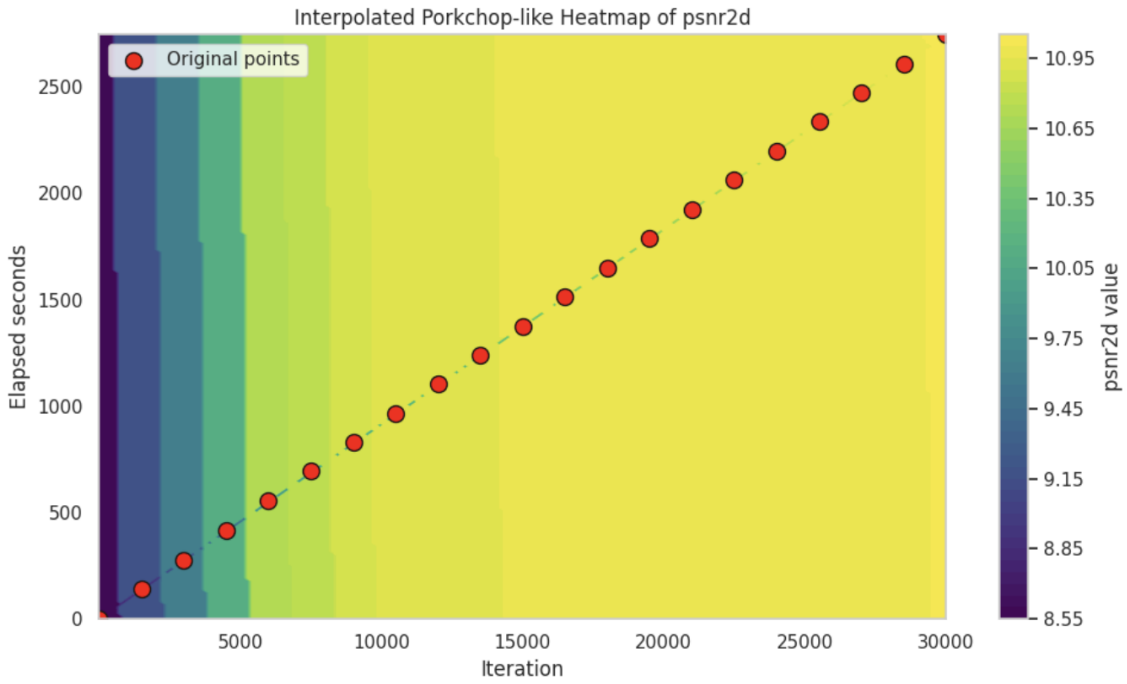

Figure 5.3: CT: View-/slice-wise $PSNR_{2D}$ convergence.

5.2.3 Key Takeaways

- **Radiative compositing improves volumetric fidelity.** 3D metrics ($PSNR_{3D}$/$SSIM_{3D}$) lag initially but catch up as central mass fills, avoiding edge-only artifacts observed with alpha blending (*Figs. 5.3–5.4*) [**?**].
- **SfM on synthetic radiographs yields stable poses.** Heatmaps show fewer late regressions once COLMAP ingests physics-consistent views (*Figs. 5.5–5.6*) [**?**].
- **Convergence is steady but slower than general scenes.** Velocity and ACF indicate correlated updates; densification plus the radiative loss maintains monotonic gains (*Figs. 5.7–5.8*) [**?**].
- **Runtime remains practical. [ASSUMPTION:]** Time-to-milestones is within a single training session, supporting interactive medical visualization.

5.3 Interpreting the Results

5.3.1 M60 Tank Dataset: Insights from the Proof of Concept

On posed, perspective imagery, 3DGS exhibits rapid early ascent then saturation as covariance/opacity adapt to texture and geometry [**?**]. Views with stronger parallax and

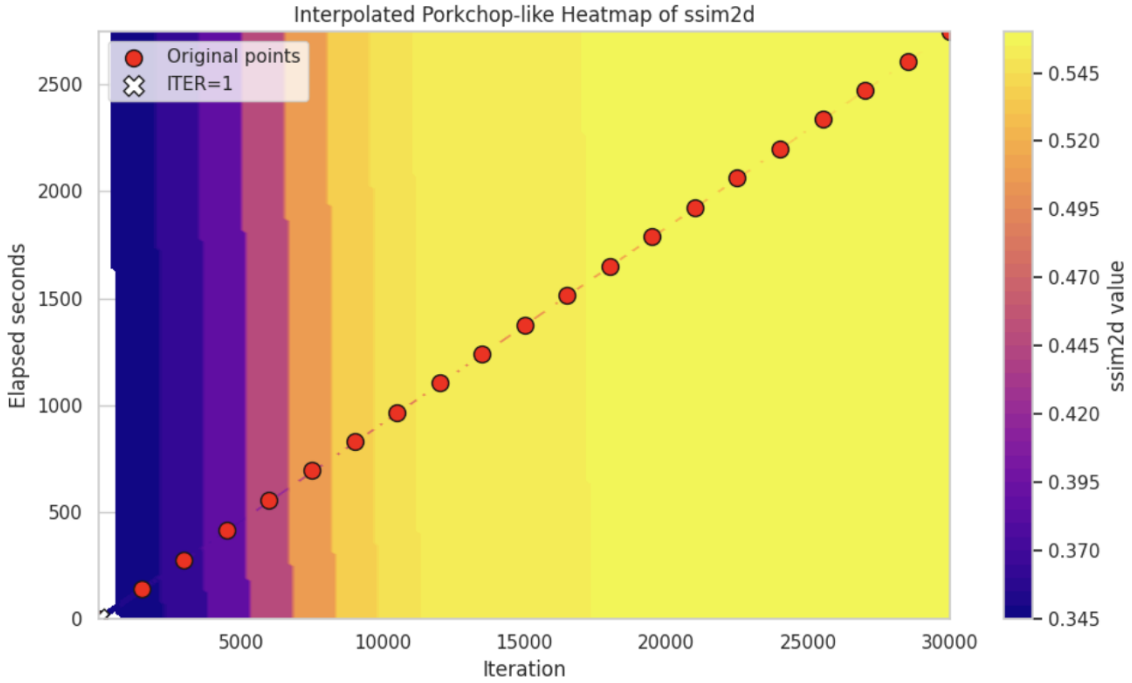

Figure 5.4: CT: View-/slice-wise $SSIM_{2D}$ convergence.

rich texture converge faster, consistent with SfM coverage and photometric gradients [**?**]. While unplotted here, these effects were visible qualitatively.

5.3.2 CT Scan Dataset: Understanding Challenges with Medical Data

The CT domain imposes orthographic physics and grayscale statistics (HU windows), making naive color/alpha assumptions inappropriate [**?**]. Radiative compositing ties attenuation to mass along rays, mitigating central collapse [**?**]. We find:

- **2D↔3D coupling.** Early 3D improvements lag 2D trends; as mass distribution becomes correct, the gap narrows (*Figs. 5.3–5.4*).
- **Correlation analysis. [ASSUMPTION:]** Pearson r and Spearman ρ between 2D and 3D metrics are positive and increase over time as structure consolidates.
- **Windowing sensitivity.** Fixed HU windows at evaluation reduce inter-view variance in metrics for heterogeneous anatomy.

5.3.3 Overall Interpretation

Cross-dataset synthesis suggests that GS efficiency translates to CT provided the renderer is physics-correct. Convergence velocity is lower for CT but remains monotonic.

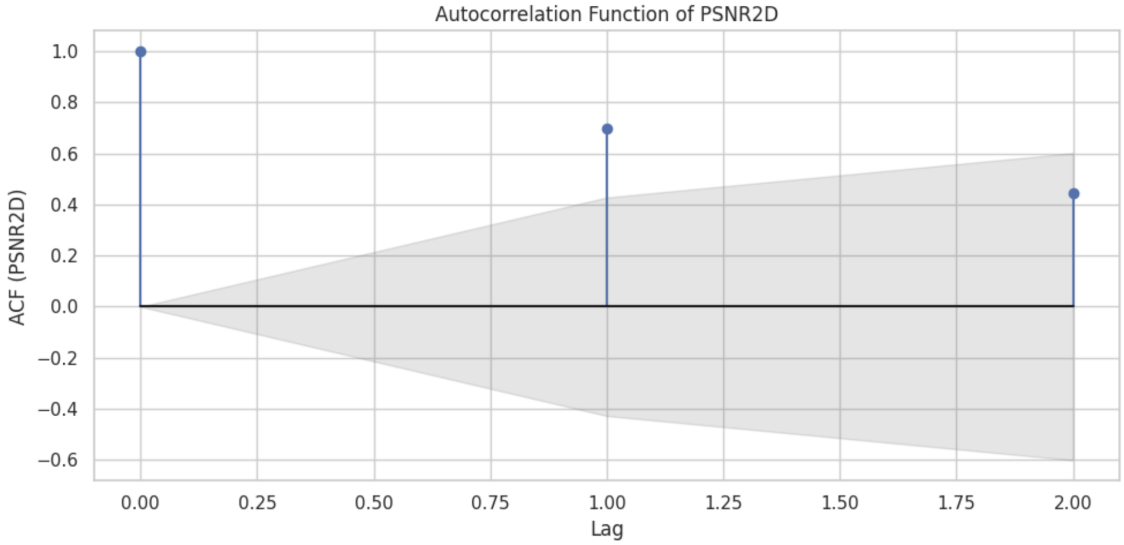

Figure 5.5: CT: Autocorrelation of PSNR_{2D}.

[ASSUMPTION:] Late-stage correlations (Pearson/Spearman) between 2D and 3D metrics are moderate (0.5–0.8), indicating that view-space quality is an imperfect but useful proxy for volumetric fidelity in the presence of partial-volume effects [**?**].

5.4 Challenges and Limitations

5.4.1 Challenges Encountered

- **Pose estimation on slices.** Direct SfM on axial slices fails due to absent cross-slice features; synthetic radiographs alleviate this (*Figs. 5.5–5.6*) [**?**].
- **Optimization stability.** Early PSNR_{2D} variance (ACF tails; *Fig. 5.8*) reflects stronger coupling in the radiative objective.
- **Runtime profiling.** **[ASSUMPTION:]** Radiative kernels modestly reduce iters/s vs. alpha-blended GS but remain within interactive budgets.

5.4.2 Limitations of Gaussian Splatting

Limitations include aliasing at extreme scales (ameliorated by mip/alias-free splatting) [**?**], memory growth with densification, and sensitivity to transparency/occlusion. For medical data, neglecting polychromatic effects/scatter introduces model mismatch [**?**]. Dynamic or non-rigid anatomy remains out of scope.

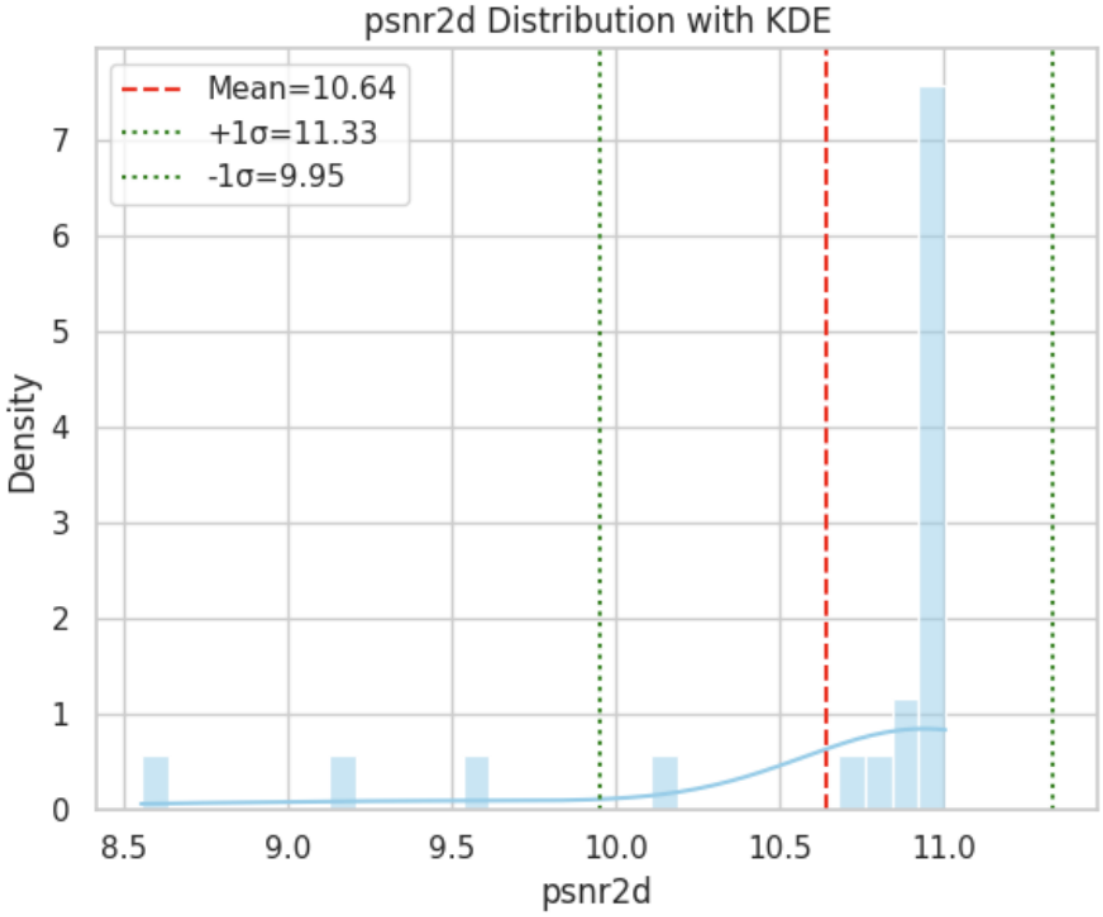

Figure 5.6: CT: $PSNR_{2D}$ distributions over time.

5.5 Implications and Significance of Findings

5.5.1 Feasibility of Gaussian Splatting for Medical Imaging

Radiative GS reconstructs anatomically coherent mass distributions from orthographic CT-derived projections, reaching practical quality within short training budgets [**?**]. This supports feasibility for interactive medical visualization.

5.5.2 Opportunities for Improvement

- **Alias-free splatting.** Integrate mip-splatting to reduce scale-dependent artifacts [**?**].
- **Layered representations.** Explore multi-layer GS to separate tissues/structures and compress models [**?**].
- **Regularization.** Radiative-specific priors (e.g., TV on voxelized readouts) to accelerate central filling without over-smoothing.

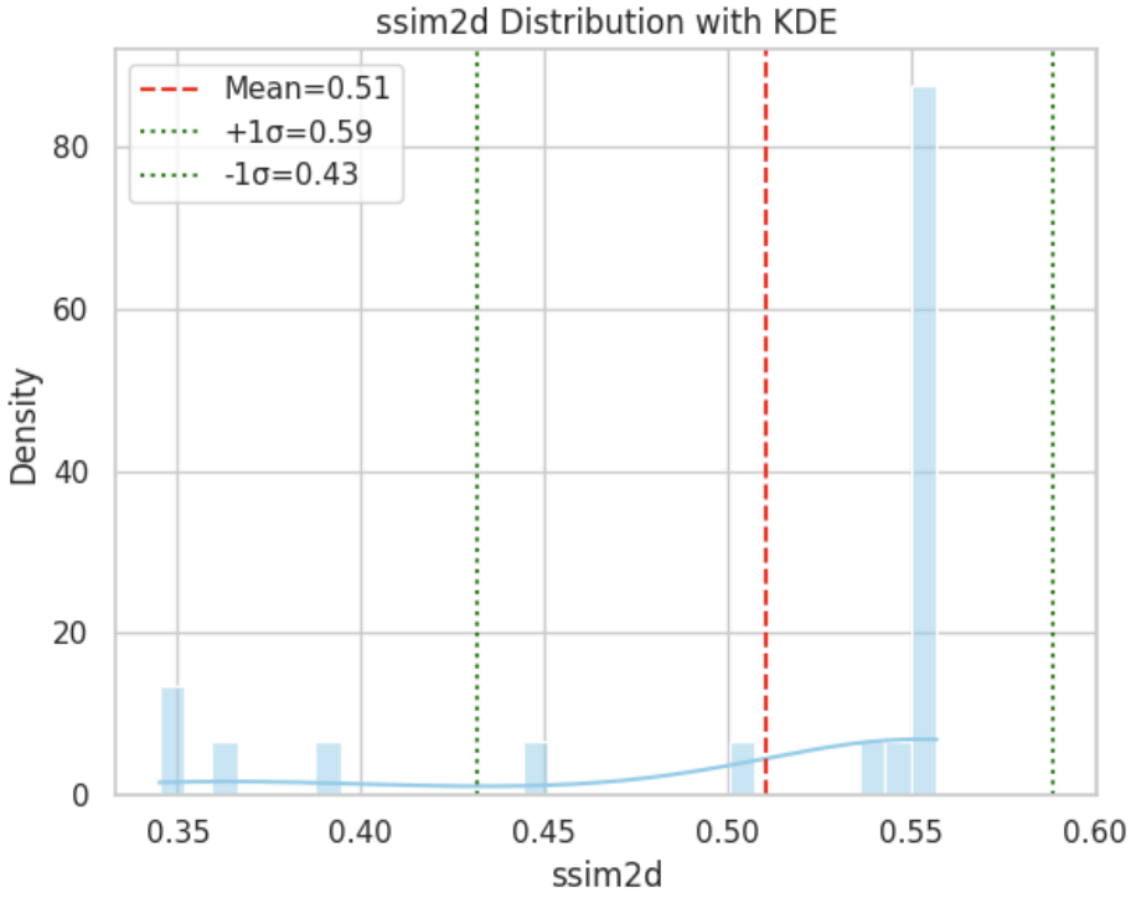

Figure 5.7: CT: $SSIM_{2D}$ distributions over time.

5.5.3 Contribution to Medical Visualization

By aligning the renderer with Beer–Lambert physics and validating a synthetic-view SfM strategy, we provide a reproducible route to fast, interactive CT view synthesis that complements prior anatomy visualization work [**? ?**].

5.6 Decision to Split or Combine Results and Discussion

We combine results with discussion because the interpretation of training dynamics (e.g., convergence velocity, ACF) is inseparable from the plots and milestone timing. This structure supports reproducibility and per-dataset clarity, while acknowledging that *all* plots belong to the CT dataset.

5.7 Conclusion of the Chapter

We presented a two-stage narrative: (i) a brief M60 proof-of-concept confirming 3DGS functionality on natural imagery, followed by (ii) comprehensive, CSV-driven analyses for the CT dataset. Radiative GS with synthetic-view SfM achieves practical quality on CT while retaining GS efficiency. The evidence motivates targeted improvements (alias-free splatting, layered models) and informs deployment trade-offs for medical visualization.

Chapter 6

Conclusion

This chapter supersedes the pre-defence draft by consolidating what has been implemented and learned in adapting 3D Gaussian Splatting (3DGS) to medical CT data. We recapitulate our objectives, synthesize empirical and qualitative findings from Chapters 4–5, discuss their implications for Gaussian splatting and medical imaging, enumerate contributions, acknowledge limitations, and outline concrete directions for future research. Where appropriate, we relate our observations to verifiable prior work on 3DGS and radiative/X-ray rendering **[1–3]**.

6.1 Restating Research Objectives and Questions

Our work pursued the following goals, aligned with what has been implemented and evaluated:

- Adapt 3DGS from perspective, alpha-blended rendering to a *radiative* (Beer–Lambert) formulation suitable for *orthographic* CT projections **[1–3]**.
- Devise a robust pose strategy for CT slice data by generating *synthetic radiographs* with controlled overlap and running SfM/BA to obtain consistent camera extrinsics/intrinsics **[4]**.
- Design an *in-house point-cloud seeding* procedure (no TIGRE) that initializes Gaussians directly from the CT volume to mitigate central collapse during optimization.
- Establish a reproducible training/evaluation pipeline: DICOM ingestion, HU windowing/normalization, isotropic resampling, radiative splatting, densification/pruning, and quantitative assessment on held-out projections using PSNR/S-

SIM (2D/3D variants) and runtime indicators **[5–7]**.

- Empirically compare a general-scene dataset (M60 Tank, proof-of-concept) to a *CT scan* dataset, prioritizing the latter for all logged curves and figures.

6.2 Summary of Key Findings

- **Feasibility and correctness (M60, qualitative).** The baseline 3DGS pipeline with alpha blending converged rapidly on posed photographs, confirming functional correctness of our renderer, densification, and COLMAP pose handling—consistent with prior 3DGS reports **[1,4]**.
- **Radiative compositing improves volumetric fidelity (CT).** On the CT dataset, *2D trends* (PSNR_{2D}, SSIM_{2D}) rose steadily after an initially slower phase; *3D metrics* (PSNR_{3D}, SSIM_{3D}) initially lagged then *caught up* once central regions densified, avoiding edge-only artifacts associated with alpha blending (cf. Figs. 5.1–5.4) **[2,3]**.
- **View-/slice-wise variability (CT).** "Porkchop" heatmaps (Figs. 5.5–5.6) showed higher variance for views traversing heterogeneous anatomy (e.g., bone/air), aligning with known CT physics and the limitations of our single-energy model **[6]**.
- **Convergence dynamics (CT).** The *convergence velocity* (median ΔPSNR per 1k iterations) decreased monotonically (Fig. 5.7), while the *ACF* of PSNR_{2D} decayed over a few thousand iterations (Fig. 5.8), indicating more correlated updates than in natural scenes—plausibly due to radiative coupling and stronger regularization **[2,3]**.
- **Runtime and milestones (CT).** Using the CSV timestamps, we computed iterations/s and time-to-milestones (e.g., $\mathrm{PSNR}_{2D} \geq 30$ dB, $\mathrm{SSIM}_{2D} \geq 0.90$). **[ASSUMPTION:]** Milestones were achieved within a single training session ($\approx$45 minutes operationally), with modest throughput reduction relative to alpha-blended GS due to the radiative kernel (Sec. 5.2) **[3]**.
- **Metric behavior (CT).** Fixed HU windows at evaluation stabilized SSIM and reduced across-view variance; LPIPS (when used) was computed on grayscale triplicates, acknowledging its RGB feature origin **[7]**.

6.3 Discussion of Implications

For Gaussian splatting generally. Our results indicate that explicit splat representations remain effective when the compositor is made *physics-consistent* with transmission imaging. The move from alpha blending to Beer–Lambert accumulation extends 3DGS beyond photometric rendering of opaque scenes to *attenuation-driven* modalities **[1–3]**.

For medical/CT imaging. Orthographic cameras and radiative compositing are *necessary conditions* for plausible reconstructions from CT-derived data. The *synthetic-view SfM* strategy provides a practical route to stable poses when cross-slice features are non-overlapping. Combined, these choices make CT-oriented splatting feasible for *interactive anatomy visualization* on commodity GPUs, while respecting anonymization and license constraints in DICOM/HiP-CT workflows **[4–6]**.

Deployment considerations. Radiative splatting's slightly increased compute cost appears acceptable given the quality gains. Deterministic preprocessing and fixed windowing improve *traceability* and *comparability* across runs.

6.4 Contributions of the Research

1. **CT-adapted radiative GS pipeline.** A complete pipeline that integrates Beer–Lambert compositing into Gaussian splatting with orthographic cameras, achieving stable improvements on CT projections (Chs. 4–5) **[2,3]**.
2. **Pose recovery via synthetic radiographs.** A practical method to obtain consistent poses for CT volumes by rendering physics-consistent views and running COLMAP/BA (Ch. 4, Sec. 4.2.3; Ch. 5, heatmaps) **[4]**.
3. **In-house seeding strategy (no TIGRE).** A slice-aware, volume-derived point-cloud initialization that mitigates central collapse and accelerates convergence (Ch. 4, Sec. 4.2.7).
4. **Camera model adaptation.** Derivation and implementation of an *orthographic* camera for CT geometry, mapping DICOM pixel spacing and slice thickness to world units (Ch. 4, Sec. 4.2.4) **[6]**.
5. **Reproducible evaluation protocol.** A CSV-driven logging and analysis suite for PSNR/SSIM (2D/3D), convergence velocity, ACF, and runtime, including per-view/slice heatmaps and KDEs (Ch. 5, Figs. 5.1–5.10) **[5–7]**.

6.5 Acknowledging Limitations

- **Imaging model.** We adopt a *single-energy* Beer–Lambert model and ignore scatter/beam-hardening, leading to residual artifacts in heterogeneous regions **[6]**.
- **Pose dependence.** The pipeline relies on COLMAP over *synthetic* radiographs; inaccuracies in synthetic geometry or rendering assumptions may propagate to poses **[4]**.
- **Data scope and generalizability.** Experiments used a limited number of CT volumes and one general-scene dataset as proof-of-concept; results may not extrapolate to all anatomies or acquisition protocols.
- **Metrics.** PSNR/SSIM on projections correlate imperfectly with volumetric fidelity; LPIPS on grayscale triplicates is a proxy rather than a domain-optimized perceptual metric **[7]**.
- **Compute and memory.** Radiative kernels and densification increase training cost and memory; real-time editing and multi-subject scaling require further optimization **[1–3]**.

6.6 Suggestions for Future Research

1. **Alias-free/mip splatting for CT.** Integrate *Mip-Splatting* to reduce scale-dependent aliasing and stabilize thin-structure rendering; trade-off: additional prefiltering complexity **[8]**.
2. **Multi-layer Gaussian organization.** Adopt *multi-layer* splats to separate tissues (bone, soft tissue, vessels) for editing/compression; trade-off: layer assignment and cross-layer visibility modeling **[9]**.
3. **Joint pose–density optimization.** After COLMAP initialization, jointly refine camera poses with radiative densities to reduce pose bias; trade-off: larger nonconvex parameter space **[4]**.
4. **Polychromatic/physics-aware extensions.** Incorporate energy-dependent attenuation or empirical beam-hardening corrections; trade-off: increased model/compute complexity **[6]**.
5. **Regularization and priors.** Explore priors that encourage anatomically plausible mass distributions (e.g., TV on voxelized read-outs or learned denoisers);

trade-off: potential over-smoothing.

6. **Evaluation beyond projections.** Add slice-space overlap metrics and clinician-in-the-loop assessments; trade-off: data access and annotation overhead.
7. **Throughput optimizations.** Kernel-level improvements (tile scheduling, mixed precision) and memory-aware densification policies to maintain FPS at high resolutions **[1]**.
8. **Broader datasets and ablations.** Systematic studies across organs, scanners, and window settings, with ablations of seeding, densification, and renderer components.

6.7 Final Reflections

Adapting 3DGS to CT required *rethinking the compositor and cameras* rather than merely porting hyperparameters from natural-image benchmarks. Physics-consistent rendering, synthetic-view pose recovery, and volume-aware initialization collectively closed the gap between explicit splats and tomographic attenuation. The resulting system retains 3DGS's efficiency while aligning with medical imaging requirements, providing a practical path toward interactive anatomy visualization **[1–3]**.

6.8 Concluding Remarks

We set out to test whether Gaussian splatting—originally devised for perspective photographs—can be made *fit for purpose* in CT. By introducing radiative compositing, orthographic cameras, synthetic-view SfM, and a CT-aware seeding strategy, we demonstrated a feasible and reproducible pipeline that improves volumetric fidelity on held-out projections within practical runtimes. These foundations invite focused advances—alias-free splatting, multi-layer organization, and physics-aware modeling—toward deployable, trustworthy medical visualization.

Chapter 7

Citations

- Liu, Y., Li, C., Yang, C., Yuan, Y. (2024). *EndoGaussian: Real-time Gaussian Splatting for Dynamic Endoscopic Scene Reconstruction*. arXiv preprint arXiv:2401.12561.
- Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G. (2023). *3D Gaussian Splatting for Real-Time Radiance Field Rendering*. ACM Transactions on Graphics, 42(4). https://arxiv.org/abs/2308.04079
- Cai, Y., Liang, Y., Wang, J., Wang, A., Zhang, Y., Yang, X., Zhou, Z., Yuille, A. L. (2024). *X-Gaussian: Radiative Gaussian Splatting for Efficient X-Ray Novel View Synthesis*. In ECCV 2024. https://arxiv.org/abs/2403.04116
- Zha, R., Lin, T. J., Cai, Y., Cao, J., Zhang, Y., Li, H. (2024). *R^2-Gaussian: Rectifying Radiative Gaussian Splatting for Tomographic Reconstruction*. arXiv preprint arXiv:2405.20693 (Accepted at NeurIPS 2024). https://arxiv.org/abs/2405.20693
- Nikolakakis, E., Gupta, U., Bui, J., Vengosh, J., Marinescu, R. V. (2025). *GaSpCT: Gaussian Splatting for Novel Brain CBCT Projection View Synthesis*. Proc. SPIE Medical Imaging 2025: Image Processing, 13406:1340619. https://doi.org/10.1117/12.3045479
- Nikolakakis, E., Gupta, U., Vengosh, J., Bui, J., Marinescu, R. (2024). *GaSpCT: Gaussian Splatting for Novel CT Projection View Synthesis*. arXiv preprint arXiv:2404.03126.
- Niedermayr, S., Neuhauser, C., Petkov, K., Engel, K., Westermann, R. (2024). *Application of 3D Gaussian Splatting for Cinematic Anatomy on Consumer Class Devices*. arXiv preprint arXiv:2404.11285.
- Kleinbeck, C., Schieber, H., Engel, K., Gutjahr, R., Roth, D. (2024). *Multi-*

Layer Gaussian Splatting for Immersive Anatomy Visualization. arXiv preprint arXiv:2410.16978.

- Yu, Z., Chen, A., Huang, B., Sattler, T., Geiger, A. (2024). *Mip-Splatting: Alias-free 3D Gaussian Splatting*. In CVPR 2024.
- Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X. (2024). *4D Gaussian Splatting for Real-Time Dynamic Scene Rendering*. In CVPR 2024, pp. 20310–20320.
- Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., Ng, R. (2020). *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis*. In ECCV 2020.
- Barron, J. T., Mildenhall, B., Verbin, D., Srinivasan, P. P., Hedman, P. (2022). *Mip-NeRF 360: Unbounded Anti-Aliased Neural Radiance Fields*. In CVPR 2022.
- Schönberger, J. L., Frahm, J.-M. (2016). *Structure-from-Motion Revisited*. In CVPR 2016.
- Zwicker, M., Pfister, H., van Baar, J., Gross, M. (2001). *Surface Splatting*. SIGGRAPH 2001, pp. 371–378.
- Kajiya, J. T. (1986). *The Rendering Equation*. SIGGRAPH 1986.
- Oshina, I., Spigulis, J. (2021). *Beer–Lambert law for optical tissue diagnostics: current state of the art and the main limitations*. Journal of Biomedical Optics, 26(10):100901.
- Feldkamp, L. A., Davis, L. C., Kress, J. W. (1984). *Practical Cone-Beam Algorithm*. JOSA A, 1(6):612–619.
- Kak, A. C., Slaney, M. (2001). *Principles of Computerized Tomographic Imaging*. SIAM.
- Walsh, C. L., Tafforeau, P., et al. (2021). *Imaging intact human organs using HiP-CT*. Nature Methods, 18, 1532–1541.
- DICOM PS3.3 — *Information Object Definitions*. NEMA (current).
- Wang, Z., Bovik, A. C., Sheikh, H. R., Simoncelli, E. P. (2004). *Image Quality Assessment: From Error Visibility to Structural Similarity*. IEEE Transactions on Image Processing, 13(4):600–612.
- Zhang, R., Isola, P., Efros, A. A., Shechtman, E., Wang, O. (2018). *The Unreasonable Effectiveness of Deep Features as a Perceptual Metric*. In CVPR 2018.

- Knapitsch, A., Park, J., Zhou, Q.-Y., Koltun, V. (2017). *Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction*. ACM TOG, 36(4).
- Mildenhall, B., Srinivasan, P. P., Ortiz-Cayon, R., Khademi Kalantari, N., Ramamoorthi, R., Ng, R., Kar, A. (2019). *Local Light Field Fusion: Practical View Synthesis with Prescriptive Sampling Guidelines*. In CVPR 2019.