Novel View Synthesis for Cinematic Anatomy on Mobile and Immersive Displays

Simon Niedermayr, Christoph Neuhauser, Kaloian Petkov, Klaus Engel and Rüdiger Westermann

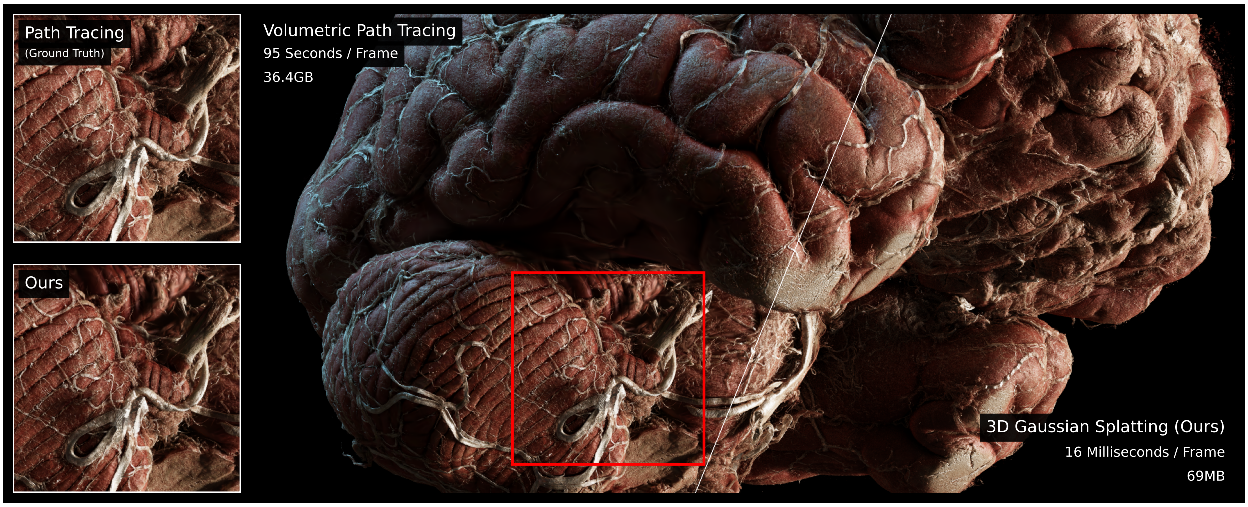


Fig. 1: Left: path-traced image of a 36 GB HiP-CT dataset, rendered in 95 seconds on a high-end GPU. Right: Gaussian splat representation of the same dataset requiring 69 MB and rendered at 60 frames per second. The rendering resolution is 2048x2048 pixels.

Abstract—
Interactive photorealistic visualization of 3D anatomy (i.e., Cinematic Anatomy) is used in medical education to explain the structure of the human body. It is currently restricted to frontal teaching scenarios, where the demonstrator needs a powerful GPU and high-speed access to a large storage device where the dataset is hosted. We demonstrate the use of novel view synthesis via compressed 3D Gaussian splatting to overcome this restriction and to enable students to perform cinematic anatomy on lightweight mobile devices and in virtual reality environments. We present an automatic approach for finding a set of images that captures all potentially seen structures in the data. By mixing closeup views with images from a distance, the splat representation can recover structures up to the voxel resolution. The use of Mip-Splatting enables smooth transitions when the focal length is increased. Even for GB datasets, the final renderable representation can usually be compressed to less than 70 MB, enabling interactive rendering on low-end devices using rasterization.

Index Terms—Novel view synthesis, cinematic volume rendering, 3D Gaussian splatting

1 Introduction

Cinematic Anatomy is an immersive anatomy learning application developed by Siemens Healthineers, which is designed to improve anatomy education through the use of photorealistic 3D visualization and cinematic rendering [12]. Instead of real 3D anatomy models it utilizes volume data provided by medical devices such as computed tomography (CT) and magnetic resonance imaging (MRI) scanners.

The application is used in the field of anatomy education, specifically for teaching the diverse and complex individual human anatomy, anatomical variations, pathology, and age-related degeneration. It aims to enhance learners' competency with immersive photorealistic 3D visualization of anatomy. In contrast to idealized anatomy models, learning with image data from real patients provides a complete experience of the patient journey from hospitalization to rehabilitation.

Additionally to stereoscopic projection modes for frontal education, Cinematic Anatomy also supports augmented and virtual reality headsets for personalized learning experiences. The portability of the created content is however often limited by the data size, especially when employing data from high-resolution imaging modalities like photon-counting CT, 7 Tesla MRI and phase-contrast CT [50]. Thus, Cinematic Anatomy is mostly used in frontal teaching scenarios, where the demonstrator uses a powerful GPU and has high-speed access to a large storage device where the dataset is stored.

We demonstrate that the limitations of Cinematic Anatomy can be addressed via novel view synthesis [35], a computer vision technique that reconstructs a volumetric radiance field of a scene from images of this scene. The reconstructed field can then be rendered with direct volume ray-casting from arbitrary views. With novel view synthesis from pre-rendered anatomy images, time consuming path-tracing can be avoided at demonstration time. On the other hand, visualization parameters such as transfer functions are baked into the volumetric radiance field and cannot be changed. We address this restriction by using compressed 3D Gaussian splatting for novel view synthesis. The resulting renderable representation has such a small memory footprint that multiple representations under different transfer function settings can be prepared and stored.

3D Gaussian splatting (3DGS) has been proposed by Zwicker *et al*. [61] to efficiently compute the projections of 3D Gaussian kernels onto the 2D image plane. Differentiable 3DGS [25] optimizes the number and parameters of the Gaussian kernels that are used to represent the scene. The Gaussians are optimized in a training process to reproduce initial images of the scene when rendered. Compressed 3DGS [38] has significantly reduced the memory consumption of the reconstructed Gaussian representation. Since it uses GPU rasterization, it runs efficiently even on mobile devices and can be integrated seamlessly into VR/AR environments. Fig. 1 demonstrates these properties with a high-resolution CT scan.

Contribution. To enable compressed 3DGS for Cinematic Anatomy, we make the following contributions:

- We present an automatic approach for finding a set of images that captures all potentially seen structures under the current transfer function setting.
- We extend 3DGS [61] with differentiable alpha channel rendering to create background-free reconstructions.
- We embed Mip-Splatting [59] to account for different levels of detail and enable smooth transitions when the focal length is increased.
- We analyze the quality, performance and memory requirements with a number of high-resolution medical datasets that are rendered using a publically available Cinematic Anatomy tool.

Our results demonstrate that even for multiple transfer function settings the overall memory requirement is significantly below of what is required by the initial dataset. For each single setting, the renderable representation is so small that it can be quickly downloaded over low-bandwidth channels and rendered on mobile devices. This enables Cinematic Anatomy with datasets that are initially so large that they do not even fit into the VRAM of high-end GPUs. Rendering performance is about two orders of magnitudes faster than optimized path-tracing, with almost no perceptible loss of image quality. This even facilitates the use of cinematic anatomy in mobile VR/AR environments with GPU multi-view rendering for stereoscopic displays.

Limitations. Besides the outlined strengths, 3DGS comes with the following limitations for Cinematic Anatomy: Firstly, lighting conditions are baked into the 3D Gaussian representation and cannot be changed during rendering. Secondly, the use of preset transfer functions hinders exploratory tasks, where domain experts might want to interactively perform changes of transfer function parameters. Thirdly, for highly transparent settings the quality of 3DGS currently decreases, requiring further research on tailored solutions to effectively cope with such scenarios.

2 Related Work

Differentiable 3D Gaussian splatting (3DGS) [25] builds upon elliptical weighted average (EWA) volume splatting [61] to efficiently compute the projections of 3D Gaussian kernels onto the 2D image plane. In addition, the number and parameters of the Gaussian kernels that are used to model the scene are optimized with differentiable rendering. Mip-Splatting [59] modifies 3DGS by integrating anti-aliasing with a 3D smoothing and a 2D Mip filter. It achieves improved quality of novel views at scales the Gaussian representation has not been optimized for. A number of approaches have concurrently proposed to convert the 3D Gaussian representations generated by 3DGS into a more compact form [27, 38]. These works have shown that for typical scenes the memory requirements of 3DGS can be brought below 50 MBytes, without any noticeable differences in the reconstructed images.

3DGS falls into the category of methods for novel view synthesis, and it overcomes in particular the difficulties of voxel-based approaches [15,44] to deal with sparsity. Even though approaches based on adaptive hash grids [37], tensor decomposition [9] or variants using dedicated compression schemes [29,42] can effectively reduce the required memory, due to the use of volume ray-casting they require high-end GPUs to achieve reasonable rendering performance. The same limitation holds for differentiable volume rendering [55], which, similar in spirit to 3DGS, optimizes optical properties on a dense voxel grid using image-based loss functions.

An alternative approach for generating a compact volumetric representation is by means of Scene Representation Networks [11, 34, 40], i.e., fully-connected neural networks which have been introduced initially to encode a surface model as an implicit 3D function. Lu *et al*. [30] demonstrate the use of SRNs for volume data compression. By overfitting a network to a volume dataset, the network learns a compact latent space representation from which the initial dataset can be reconstructed. This, however, comes at the expense of subsequent network evaluations during rendering. This makes even the performance of GPU-friendly implementations [53] fall significantly below the performance of 3DGS.

While 3DGS uses rendered images of a volume dataset to build a renderable 3D object representation for novel view synthesis, light-field rendering [18,28] aims at generating novel views by interpolation between given images. Light-field rendering has been used to accelerate novel-view synthesis for volume data [5,8,49], yet it is fair to say that it cannot compete with 3DGS with respect to quality and memory requirements.

To generate the images that are used by 3DGS, we use Cinematic Rendering, a volumetric Monte Carlo path tracing algorithm developed by Siemens Healthineers. It is integrated into the teaching application Cinematic Anatomy, which was developed in collaboration between the Johannes Kepler University Linz and Siemens Healthineers. A first prototype was installed at the Ars Electronica Center in Linz in 2015 in the Deep Space 8k digital experience space [14], a 16-by-9-meter wall projection space providing stereoscopic 8k resolution experiences to more than 100 visitors at a time. Since then, regular interactive visualizations of clinical volumetric DICOM data have been made available to the public. Furthermore, it has been incorporated into anatomy lectures for medical students and staff. In 2021, the JKU medSPACE, a lecture space for teaching anatomy, located at the Johannes Kepler University in Linz was opened. Developed and implemented by the Ars Electronica Futurelab, the JKU medSPACE shows anatomy in quadruple stereoscopic 3D 4K projection at 14×7 meters. It utilizes Cinematic Anatomy to provide virtual anatomy education to medical students and staff. Several studies have shown the benefits of using photo-realistic volumetric rendering of clinical volume data for teaching and understanding anatomy [3, 4, 17, 43].

While our work builds upon 3DGS to interactively perform Cinematic Anatomy, other works have previously attempted to improve the performance of path tracing via image denoising [20–22], photon mapping [60], illumination caching [62] and adaptive temporal sampling [33]. Despite achieving remarkable performance gains at high quality, it is worth noting that these approaches cannot overcome the major limitation of interactive Cinematic Anatomy, i.e., it requires a rendering system with enormous memory resources to host high-resolution medical scans, as well as huge amounts of computing power to perform ray tracing with such data.

3 Cinematic anatomy pipeline

The different stages of the proposed pipeline for cinematic anatomy are shown in Fig. 2. It starts with reading a medical dataset, for which then one or multiple so-called presets are selected by the user. A preset includes the transfer function setting as well as material classifications and fixed clip planes that are used to reveal certain anatomical structures. For each preset, a set of views capturing all potentially seen structures in the data at varying resolution are computed (cf. Sec. 4.1). In this way, also structures which are not seen when generating images with camera positions on a surrounding sphere are recovered in the final object representation. These views are handed over to a physically-based renderer, i.e., a volumetric path tracer, which renders one image for every view using the corresponding preset (cf. Sec. 4.2). Once all images for a selected preset have been rendered, 3DGS is used to generate a set of 3D Gaussians with shape and appearance attributes so that their rendering matches the given images. Once the Gaussians are computed via differentiable rendering, they are compressed using sensitivity-aware vector quantization and entropy encoding (cf. Sec. 4.3). The final compressed 3DGS representation is rendered with WebGPU using GPU sorting and rasterization of projected 2D splats, with a pixel shader that evaluates and blends the 2D projections in image space.

4 Methods

4.1 View selection

Our approach iteratively generates a sequence of different viewpoints, so that every new viewpoint can recover as many as possible voxels which have not (or not sufficiently) been covered so far. While we use

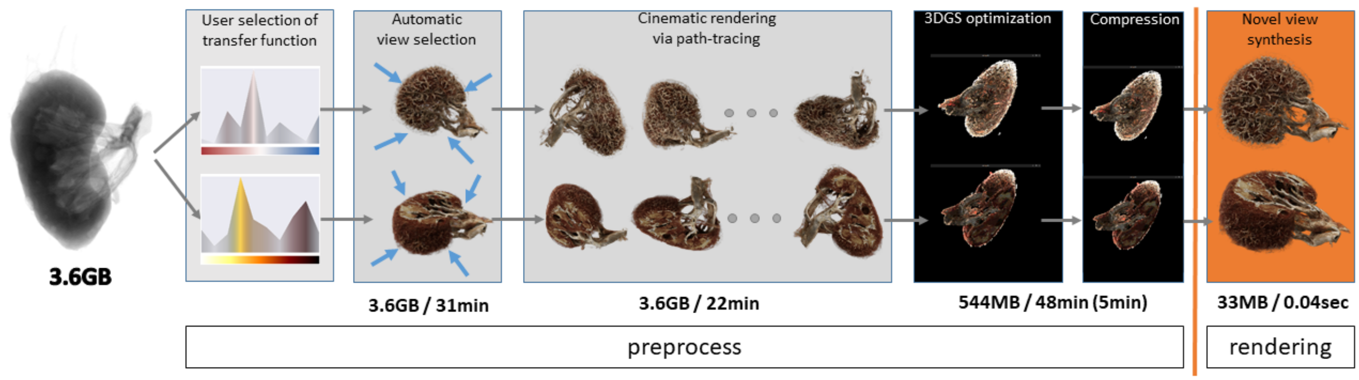


Fig. 2: Cinematic anatomy pipeline for a $1510 \times 1706 \times 1415$ HiP-CT dataset requiring 3.6 GB of memory. Numbers below each stage indicate the required memory at this stage and its computation times. 3D Gaussian splatting (3DGS) optimization uses 99 path traced images, and first generates the raw Gaussian representation in 48 minutes before it is compressed to 33 MB in 5 minutes.

this approach in a fully automated manner in this work, in practice an initial set of views would be placed uniformly around the object and additional views would be recommended and accepted or denied by a domain expert.

Our approach shares similarities with techniques that automatically optimize visualization parameters to find a *single* best viewpoint, using image-based loss functions to guide an optimizer toward this optimum [10, 23, 46–48, 58]. Viewpoint quality metrics for isosurface rendering were developed by Takahasi *et al*. [45] and Marsaglia *et al*. [31], based on applying entropy to either the rendered images or derived fields such as depth. Bordoloi *et al*. [6] introduce the use of per-voxel significance measures to quantify image entropy. Different viewpoint search algorithms using a given number of views on a sphere surrounding the volume were investigated in [32]. While these approaches first generate many possible viewpoints and select the best one using sampling, Weiss and Westermann [54] embed the optimization into the volume rendering process and let the optimizer converge to the best viewpoint.

Once a preset is selected, our method starts with computing an occupancy volume indicating for every voxel whether it contains material or not. A per-voxel visibility volume is initialized to zero. In the first phase, a set of camera positions (always looking at the center of the dataset) are generated on an ellipsoid entirely containing the dataset. The camera is moved toward or away from the center using a random offset. For each camera pose, for every non-empty voxel the transmittance along a view ray to this voxel is computed, and the per-voxel visibility is updated with the maximum of the current value and the transmittance.

The second phase generates additional views that cover as many as possible previously unseen structures, i.e., it selects camera poses that maximize the voxel visibility *gain*, which is defined as the increase of the visibility values by the transmittance values that are computed for a camera pose. We use Bayesian optimization [16, 36] for generating the new poses, which solves the optimization problem $\max_{\theta \in D} f(\theta)$ for a black box function $f : D \to \mathbb{R}$ with domain D. In our setting, f is the function that maps the camera parameters to the voxel visibility gain. The goal of Bayesian optimization is to minimize the number of function evaluations. For this, a probabilistic (usually Gaussian) surrogate model and an acquisition function are applied. The former expresses Bayesian believe about the output of the objective function derived from prior evaluations, while the latter is used for selecting the next set of parameters for evaluation of the objective function. We make use of a publically available Python library [39] for performing the optimization process. For the acquisition function, the Upper Confidence Bound (UCB) [7] is used with parameter $\kappa = 10$.

We iteratively generate additional camera poses by initializing the Bayesian optimizer with $K = 128$ randomly sampled candidate camera poses in each iteration. A candidate camera pose is discarded and re-sampled if the bounding box of the data lies outside the view frustum or the camera position is inside an occupied voxel. Then, Bayesian optimization is started to iteratively sample another set of K candidate camera poses. Here, it utilizes the prior that has been captured through previous camera poses to guide the optimization process. Only the best candidate pose is finally selected per iteration.

4.2 Image generation

We render volume data with Monte Carlo volume path tracing [2, 2, 12, 13, 24, 26] from multiple views to generate a set of training images. Rendering starts by tracing paths from the detector pixels of a virtual camera into the scene. A transfer function is used to map the scalar quantities of the 3D scans to an emissive color and density. Delta tracking [56] is applied to decide whether at a sample point a scattering event, absorption event or null collision occurred. In the case of an absorption event, the path is terminated and the emissive color is regarded as the contribution of the path. In the case of a scattering event, the Henyey-Greenstein phase-function [19] is applied to determine the next direction along which the ray is followed. In the case of a null-collision, the path is followed unchanged.

Additionally to the semi-transparent volumetric component, opaque isosurfaces can be embedded into the rendering process. For this, the local density gradient magnitude at each sampling point is computed, and an intersection with a surface is assumed when the gradient magnitude exceeds a user-specified iso value. The iso surface can also be colored by applying a transfer function that maps density to color. Global shading is computed by generating a reflection event for the surface with the new ray direction being sampled proportional to the probability density function (PDF) of a chosen bidirectional reflectance distribution function (BRDF) of the surface. This process is repeated until the ray leaves the volume domain or an absorption event happens. Absorption may be determined by a fixed maximum number of scatter events or using Russian roulette.

High dynamic range light maps are employed to look up lighting information contributions from the environment. We also support synthetic local light sources. Next event estimation is used to importance sample rays towards the light source and potentially reduce the variance of the rendered image. All Monte Carlo samples are accumulated and averaged in a floating-point accumulation buffer. A tone-mapping pass maps the accumulated result into the final lower dynamic range output buffer. For fast image generation, we apply performance optimization methods such as empty-space skipping and memory coherent scattering. The latter optimization ensures that rays of neighboring pixels are scattered in the same direction, thus ensuring optimized cache utilization.

4.3 Compressed Differentiable 3D Gaussian Splatting

With differentiable 3DGS, the object is described by a set of 3D Gaussians

$$G(x) = \alpha e^{-\frac{1}{2}x^T\Sigma^{-1}x}. \tag{1}$$

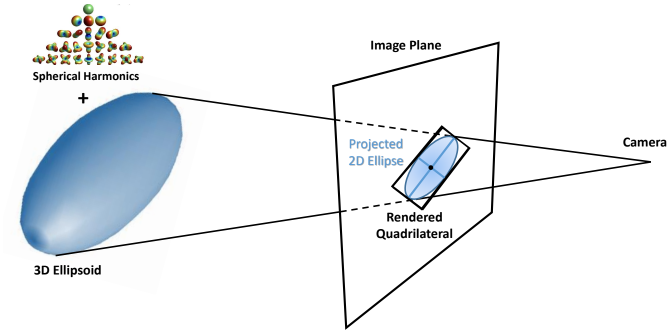


Fig. 3: Illustration of Gaussian splatting. Each 3D Gaussian has an opacity and a set of SH coefficients, to evaluate the per-pixel opacity and view-dependent color of the Gaussian's 2D projection.

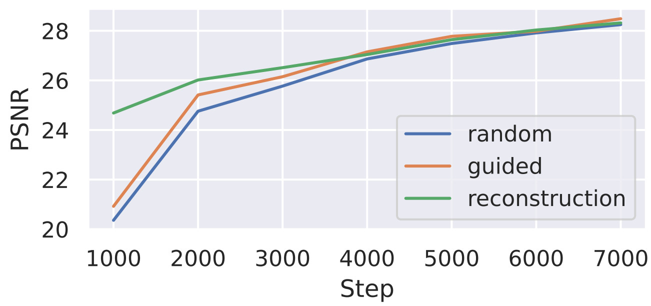


Fig. 4: Test image reconstruction using different initialization schemes. Experiments were performed with Fullbody.

Each Gaussian is centered at $x \in \mathbb{R}^3$, and a covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$ is used to describe its orientation and shape. A Gaussian has an opacity $\alpha \in [0,1]$, and a view-dependent color that is represented by a set of spherical harmonics (SH) coefficients. For each view, SH coefficients are multiplied with the view direction to obtain the color [25].

The 2D projection of a 3D Gaussian is a 2D Gaussian with covariance

$$\Sigma' = JW\Sigma W^T J^T, \tag{2}$$

where W is the view transformation matrix and J is the Jacobian of the affine approximation of the projective transformation. By computing the exponential decay of color and opacity across the 2D projection, at each pixel the color and opacity that is seen along the ray of sight through the 3D Gaussian can be reconstructed (see Fig. 3). A final pixel color C is then computed by blending the contributions of all Gaussians in sorted order:

$$C = \sum_{i\in N} c_i \alpha_i \prod_{j=1}^{i-1} (1-\alpha_j). \tag{3}$$

Here, N is the number of Gaussians affecting a pixel, and c_i, α_i, respectively, are the view-dependent color of a Gaussian and its opacity, modulated by the exponential falloff from the projected Gaussian's center point.

While Zwicker *et al*. [61] model a 3D scalar field via a set of 3D Gaussians so that the field can be reconstructed sufficiently well, Kerbl *et al*. [25] optimize the position, shape, opacity and SH coefficients of each 3D Gaussian so that their rendering matches a set of initial images of the object. The optimization is performed via differentiable rendering, by taking into account the changes in pixel color due to changes of the 3D Gaussian parameters. In the optimization process, initially selected 3D Gaussians are removed (if no contribution), adaptively split and their shapes and appearance attributes are modified to minimize an image-based loss function.

4.3.1 Compression

Due to the use of per-splat SH coefficients and shape attributes, the 3D Gaussian splat representation can become memory consuming and take up hundreds of MB or even GB per scene. Niedermayr *et al*. [38] have proposed strategies for compressing this representation. SH coefficients and Gaussian shape parameters are encoded into compact codebooks via sensitivity-aware vector quantization. To reduce the error introduced by vector quantization, the parameters are then fine-tuned on the training images after compression. To represent the scene parameters with fewer bits, quantization-aware training [41] is used. The final set of 3D Gaussians is linearized along a space-filling curve in the 3D domain. In this way, spatial coherence is maintained in the linearized sequence, and entropy encoding can be used to further compress the sequence of splat parameters.

For volume reconstruction, we have modified the compression scheme in the following way: Firstly, since Cinematic Anatomy requires the entire dataset in focus, we remove the scaling factor that is stored per Gaussian to represent scenes with objects in focus (small Gaussians) and surrounding background (large Gaussians). We call this strategy high-rate-compression (HR-compression). Since quantization of SH coefficients and shape parameters can introduce compression errors, we provide a variant that compress the final Gaussian representation at no perceivable loss in reconstruction quality. Therefore, we only use quantization-aware training to reduce all scene parameters but the Gaussians' positions to an 8-bit representation during optimization. Positions are encoded using 16 bit floating point numbers. We will subsequently call this strategy high-quality-compression (HQ-compression). On average, it gives a compression factor of 26$\times$ compared to the uncompressed 3DGS representation, bringing all scenes bellow 170 MB.

4.3.2 Volume Guided Initialization

When using 3DGS, an initial set of 3D Gaussian kernels is first selected. These Gaussians are then removed, split or re-positioned, and the shape and appearance of the Gaussian kernels is optimized. Kerbl *et al*. [25] obtain the initial positions of the 3D Gaussians from the given images with structure from motion, or with random initialization were Gaussians are randomly positioned in the scene. For volume rendering, we randomly place Gaussians within the volume bounding box and set their initial color to grey. All other parameters are initialized as proposed by Kerbl *et al*. [25].

Since in Cinematic Anatomy the 3D object and presets are known, an interesting question is whether the optimization process can be accelerated by initially placing Gaussians at locations were they will end up anyway. Thus, we initially position one Gaussian at every non-empty voxel in a low resolution version of the volume, and set the Gaussians' initial colors and opacities via the transfer function. Regions that are under-sampled by the initial sampling will be nevertheless represented by Gaussians due to adaptive splitting and relocation during optimization.

In Fig. 4 we exemplarily compare the effectiveness of the different initialization schemes based on optimization convergence for one of our test datasets. An initialization with the Gaussians' positions and colors from a previous reconstruction is used as gold standard. As can be seen, while all initialization techniques reach the same level of fidelity, volume-guided initialization does so with less iteration steps. However, it is fair to say that in all of our experiments the performance improvements were overall not significant, so that we decided to use random initialization in all upcoming tests.

4.3.3 Alpha Channel Reconstruction

In contrast to classical novel view synthesis, where only RGB colors are reconstructed, in volume rendering applications also the per-pixel accumulated opacity (i.e., alpha) needs to be reconstructed for blending correctly over the background. We extended 3DGS to allow for the differentiable rendering of images with alpha channel. Similar to Kerbl *et al*. [25] we use a combination per pixel L1 and SSIM Loss to faithfully reconstruct the alpha channel of the training images. This greatly improves the reconstruction quality as we show in Sec. 5.4.

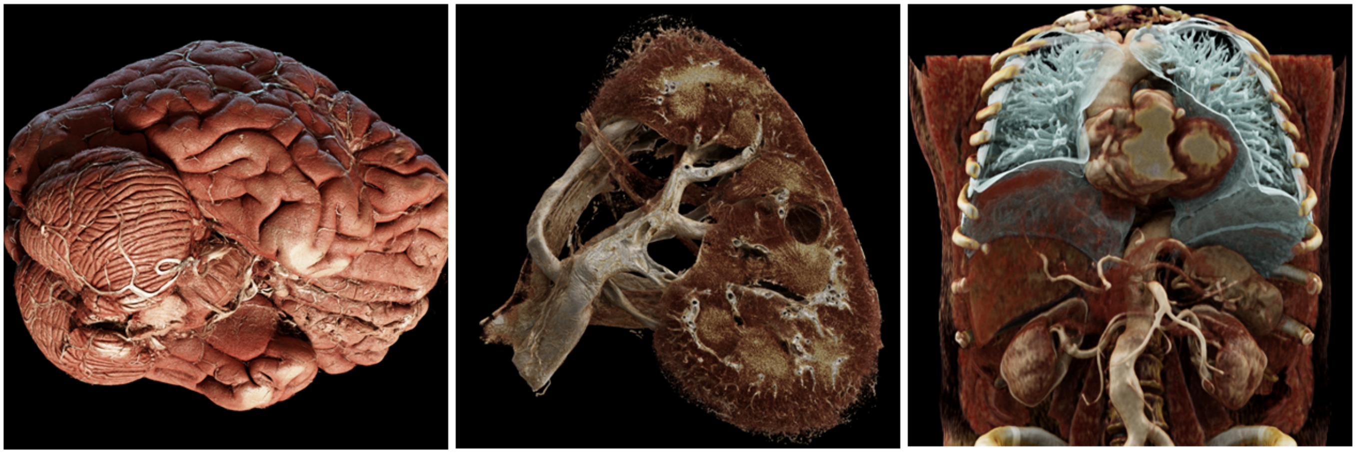


Fig. 5: From left to right, Brain ($3224 \times 3224 \times 3585$), Kidney ($1510 \times 1706 \times 1415$) and Fullbody ($317 \times 317 \times 835$). All images rendered at full HD with HR-compressed 3D Gaussian splatting using < 70 MB per dataset at > 30 frames per second.

4.3.4 Mip Splatting

Scenes rendered with 3DGS can show severe artifacts when novel camera perspectives diverge from those the 3D Gaussian representation was optimized for. Yu *et al*. [59] name the following two reasons for this behavior: Firstly, the 3D Gaussian representation exhibits frequencies that are too high to be faithfully reconstructed by the used sampling rate. Secondly, during splat-based rendering, a 2D dilation filter is applied that causes artefacts when zooming out and 2D splats become too small.

The problem is mitigated by introducing a 3D smoothing (i.e., low-pass) filter which constrains the size of the 3D Gaussians based on the maximal sampling frequency induced by the input views. A 2D Mip filter is applied in image space to avoid under-sampling. We observe that this extension to 3DGS significantly improves the fidelity of the reconstructed volumes for varying zoom levels.

5 Results

We analyze the performance, memory consumption and quality of the proposed pipeline for Cinematic Anatomy with a variety of high-resolution medical datasets showing different anatomical structures.

Our 3DGS implementation is a modification of the code implementation provided by Kerbl *et al*. [25]. For compression and rendering, we use the settings described in [38] and their publicly available WebGPU renderer.

5.1 Datasets

The hierarchical phase-contrast tomography (HiP-CT) data was acquired at the European Synchrotron Radiation Facility (ESRF) in the context of the Human Organ Atlas project [51][1].

Kidney is a HiP-CT scan from beamline 5 of the complete left kidney from body donor LADAF-2020-27 downsampled to 50.16 µm resolution ($1510 \times 1706 \times 1415$ voxels in size) and quantized to 8 bit precision.

Brain is a HiP-CT scan from beamline 18 of the complete brain of body donor LADAF-2021-17 downsampled for rendering to 46.84 µm resolution ($3224 \times 3224 \times 3585$ voxels in size) and quantized to 8 bit precision. While the kidney data set is publically available, the brain data has not been published yet.

Fullbody is a human CT angiography scan at resolution $317 \times 317 \times 835$ from collection [52], image id *s0287*. The dataset contains some semi-transparent material that shows significant differences under directional lighting. It is rendered under complex lighting conditions to demonstrate reconstruction quality also in this situation.

[1] https://human-organ-atlas.esrf.eu

Table 1: Memory and preprocessing statistics using HR-compression. Timings given for 2048×2048 training images.

	Path Tracing			3DGS		
	Size	Time	Views	Size	Time	Gaussians
Brain	36.4 GB	158 Min	99	69 MB	106 Min	4.8 M
Kidney	3.6 GB	6 Min	101	33 MB	53 Min	2.3 M
Fullbody	0.2 GB	23 Min	99	7 MB	50 Min	0.9 M

All datasets are shown in Fig. 5. For each dataset, between one and three rendering presets have been used, which include segmentations, the selected transfer function and lighting conditions. 3DGS optimization has been performed training images of resolution 2048×2048.

5.2 Preprocessing

With a GPU providing sufficient RAM, the initial images of all datasets can be generated with the publically available Cinematic Anatomy package[2], e.g. on an NVIDIA RTX 8000/A6000 GPU with 48GB RAM and using the built in animation system to generate the views. All images used as training and test data in our experiments have been rendering on a research version providing batch rendering support, running on an NVIDIA A100 GPU for the Brain data and NVIDIA RTX A5000 for the Kidney and Fullbody data. Note that rendering HipCT in its original resolution on a single GPU requires bricking the data into multiple smaller blocks. This increases the rendering times drastically, since path tracing requires streaming each block multiple times from the CPU to the GPU.

Tab. 1 shows in columns *Size* the size of each dataset in GB compared to the size of the final Gaussian representation in MB, when compressed using HR-compression. Column *Views* shows the number of training images used for differentiable Gaussian splatting optimization. Columns *Time* show the times to render the initial images via path tracing, and the computation times for generating the compressed Gaussian representations. Note that 90% of the latter time are required by the optimization to generate the 3D Gaussian representation, and only about 10% are consumed by the compression. Column *Gaussians* gives the number of 3D Gaussians in the final representation. Note that for the body scan, the used preset uses a lung mask-dependent and clip-plane dependent transfer function, which slows down the rendering considerably. As can be seen in columns *Size*, the compressed Gaussian representation is so small that it can be downloaded over low-bandwidth channels and rendered on mobile devices equipped with mid- or even low-end GPUs.

[2] https://siemens-healthineers.com/cinematic-anatomy

5.3 View Selection

Automatic view selection is demonstrated with Fullbody, which exhibits a lot of structures which are not visible from cameras placed on an ellipsoid around the volume. As a baseline we reconstruct the volume with images from 256 randomly placed cameras on the ellipsoid. For comparison, we reduce this number to 128, and generate 128 additional cameras with the proposed view selection algorithm (see Fig. 6). As can be seen, overall improved reconstruction quality of parts not seen with random camera selection is achieved.

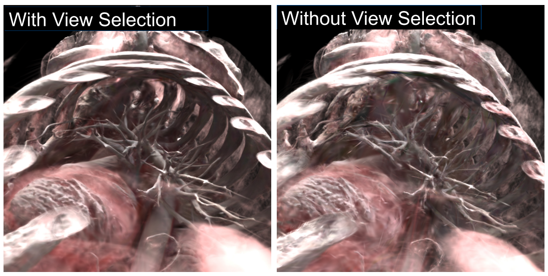


Fig. 6: Our view selection algorithm generates cameras covering unseen parts of the volume like the inside of the rib cage. The renderings show the improvement in reconstruction.

5.4 Quality Evaluation

To support a qualitative analysis, Fig. 9 and Fig. 10 compare test images that have not been seen during 3DGS optimization to the same views rendered with HQ-compressed and HR-compressed 3DGS. Close-up views emphasize that only very subtle color shifts between path traced images and images generated via HR-compressed 3DGS can be observed. HQ-compression leads to an increase in memory of a factor of three, yet differences in image quality are further reduced and become so small that they are hardly noticeable by eye.

To quantify these differences, Tab. 2 shows the average SSIM and PSNR between the test images and the novel views rendered with HR-compressed 3DGS.

Table 2: Quantitative evaluation of HR-compression, using 2048×2048 training images and averaged over all presets.

Scene	SSIM	PSNR	PSNR (Alpha)
Brain	0.72	23.23	34.09
Kidney	0.84	25.80	30.20
Fullbody	0.87	26.90	29.57

For PSNR and SSIM only pixels which are not empty (alpha > 0) in the rendered and ground truth image are considered. PSNR (Alpha) measures the PSNR for the alpha channel between rendered images and ground truth.

It is worth noting here, that significant losses in reconstruction quality are introduced when differentiable 3DGS is used for solely optimizing RGB color (see Fig. 7 for an example). As can be seen, extending 3DGS so that also opacity is considered in the optimization process improves greatly the reconstruction quality and removes unwanted artifacts caused by the background.

In a final experiment we shed light on the capabilities of 3DGS to reconstruct semi-transparent regions in a dataset. Fullbody, with a corresponding preset, is used here as an example. The preset has been selected so that certain tissue types in the dataset become semi-transparent. The bottom images in Fig. 10 show the application of this preset, including a test image and the corresponding novel views generated with compressed 3DGS. It can be seen that overall the novel view matches the test images fairly well. When looking at the closeup views,

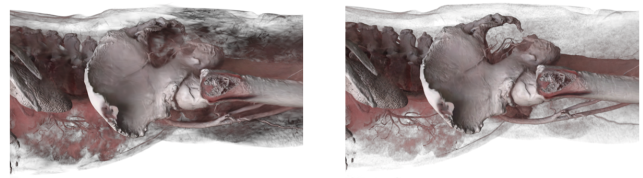


Fig. 7: Color-only reconstruction (left) leads to reconstruction artifacts, which disappear when 3DGS is optimized for color and opacity (right).

however, one sees that some small details are not reconstructed accurately, and that especially the semi-transparent structures are blurred out in the final images.

To further shed light on this situation, we have also experimented with a setting with strong directional lighting from the used environment map. In such a setting, one observes some high-frequency illumination variations especially in the volumetric regions, making it more difficult for 3DGS to accurately recover the structures. Interestingly, Fig. 8 demonstrates that the reconstruction of illumination variation works very well and does not show any severe reconstruction artefacts. At the same time, the semi-transparent regions are again blurred out to a certain extent. We believe that 3DGS has in particular problems with settings where the view rays accumulate matter over a long distance through semi-transparent, yet heterogeneous regions. In such situations, a subtle change of the camera pose can lead to strong changes of the per-pixel accumulated colors and opacities. Thus, 3DGS needs to be optimized for a significantly increased number of parameters, requiring far more Gaussians to accurately represent the data.

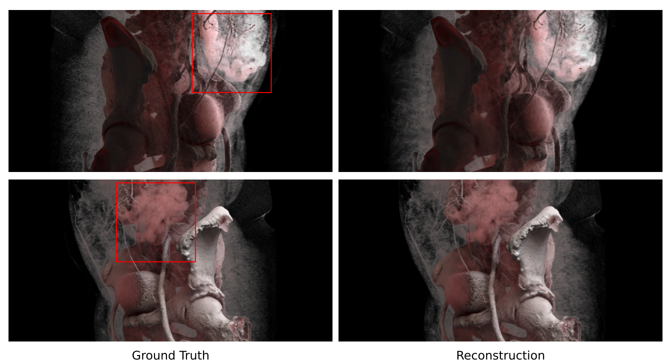


Fig. 8: Complex view-dependent lighting effects are well preserved by compressed 3DGS, even for semi-transparent material. The tissue marked with a red box shows high-frequent color variation under different perspectives.

5.5 Rendering Performance

In a pre-render pass on the GPU, view frustum culling of Gaussians is performed, and the remaining Gaussians are depth-sorted to enable order-dependent blending. This pass consumes roughly 10% of the entire rendering time. The implementation by Niedermayr *et al.* [38] uses the GPU Onesweep algorithm by Adinets and Merrill [1] for sorting. The radix sort implementation computes digit histograms and - via (chained) prefix sums - the global and bin-relative offsets of each digit in the final output. The implementation is tailored for sorting large sets of keys, and it works in-place in GPU memory. Gaussians are rendered as screen-aligned quadrilaterals, and a pixel-shader evaluates the Gaussians' colors and opacities. Of the rendering time roughly 30% and 70% are respectively devoted to sorting / geometry processing (including rasterization) and fragment processing in the pixel shader.

Tab. 3 shows that the rendering times even on an integrated iGPU is higher than 10 frames per second for the biggest dataset Brain. On current mid- to high-end GPUs, 60 frames per second can be achieved for all datasets. This makes the Cinematic Anatomy pipeline especially

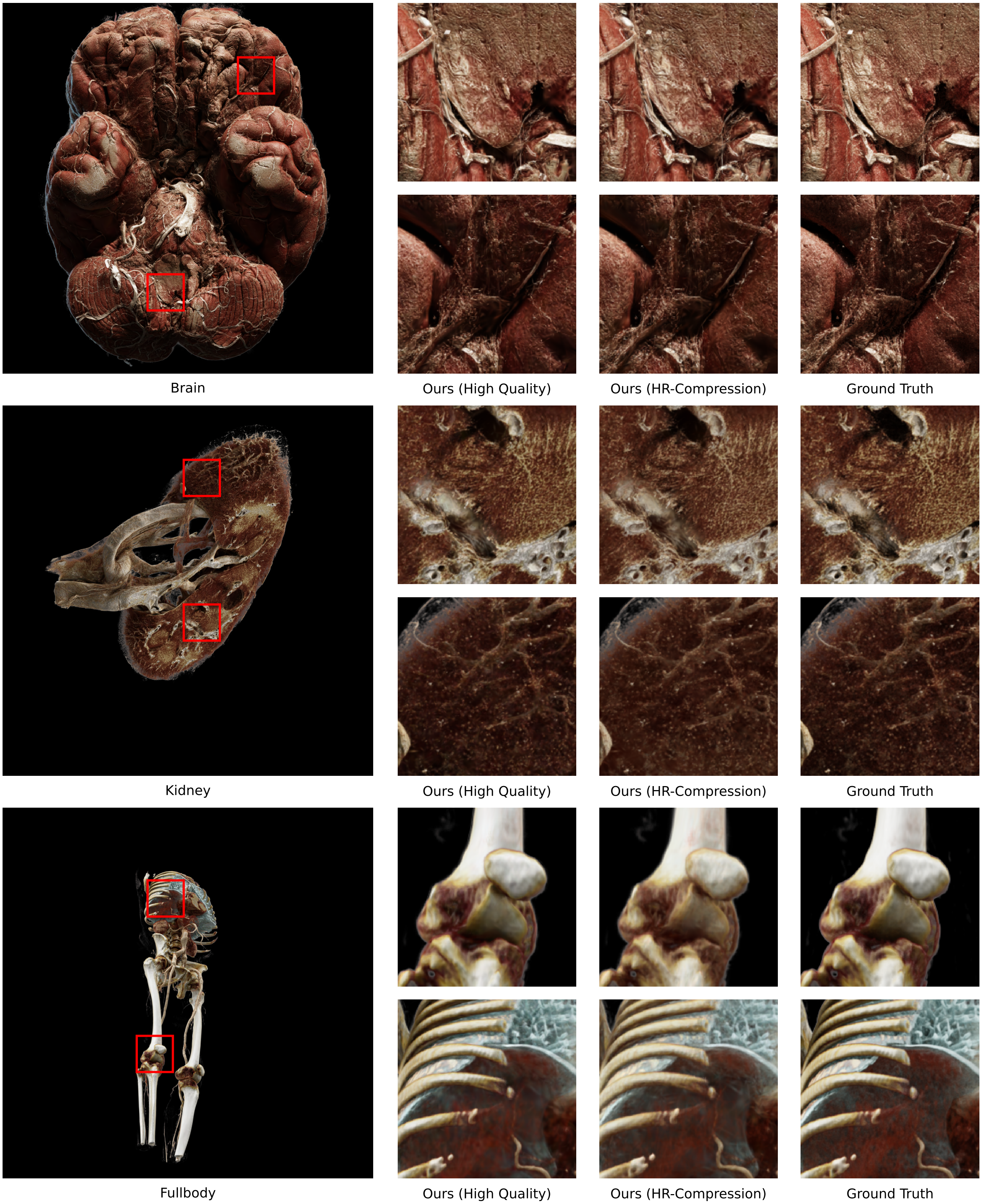


Fig. 9: Quality comparison for HQ-compressed and HR-compressed Gaussian representations. All images are from the test set.

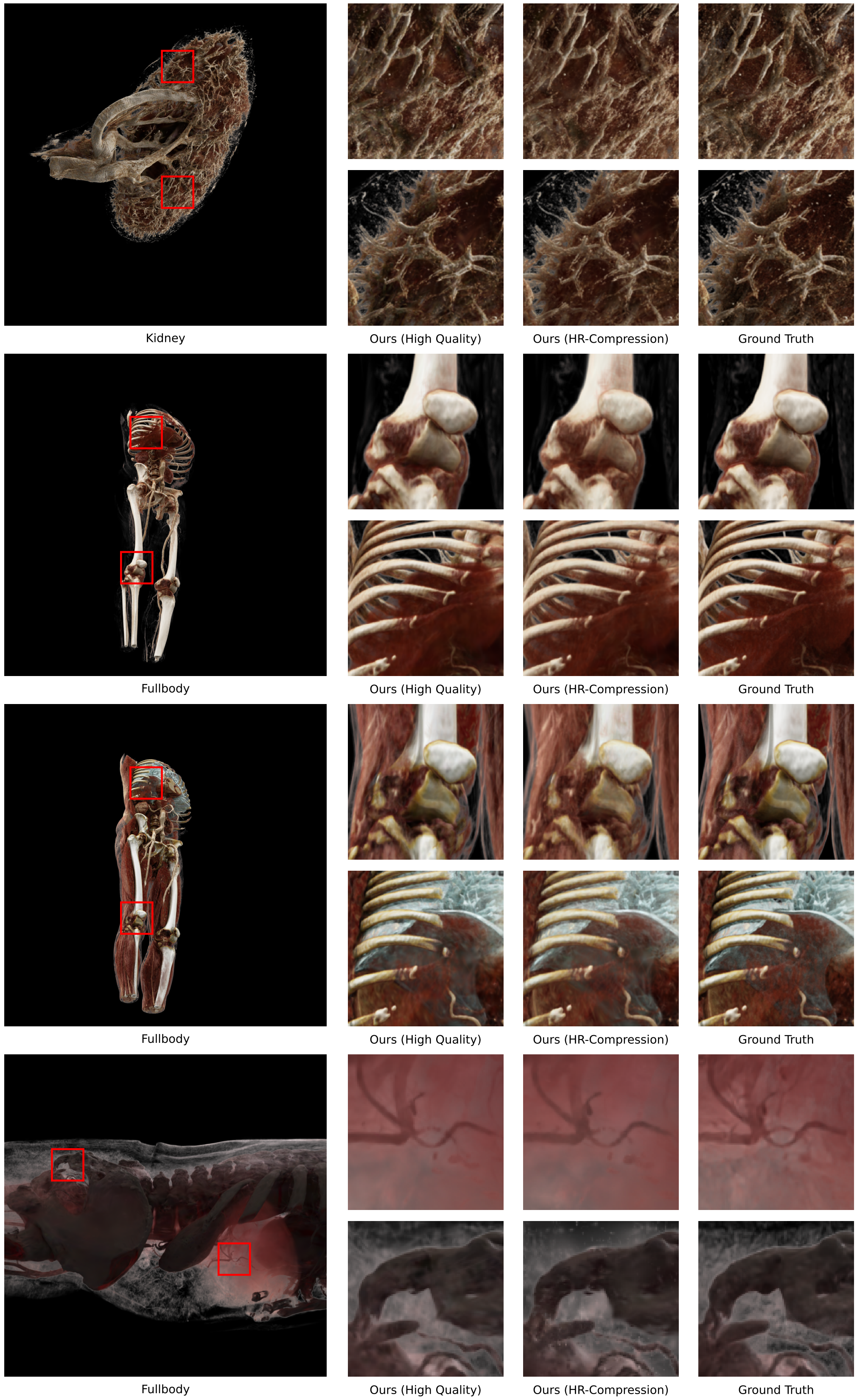
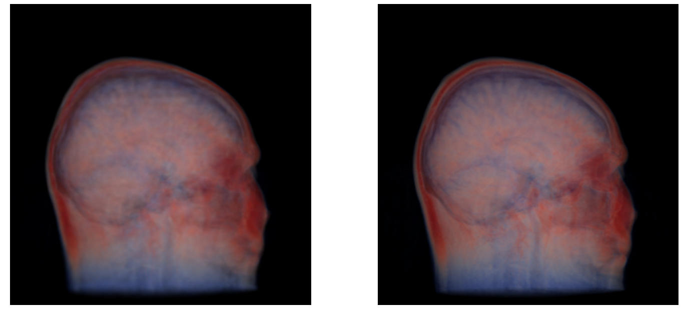


Fig. 10: Quality comparison for HQ-compressed and HR-compressed Gaussian representations. All images are from the test set.

Brain | Ours | Ground Truth

Fig. 11: With a headlight, 3DGS faces problems in some places to accurately reconstruct structures with high-frequent changes in illumination.



(a) Reconstruction | (b) Ground Truth

Fig. 12: Compressed 3DGS with a semi-transparent skull dataset. Fine details are blurred out in the reconstruction.

appealing for applications where stereoscopic rendering is required. While low memory consumption facilitates efficient rendering on mobile devices, for instance, in mobile AR applications, high rendering performance is required to render two images (one for the left and one for right eye) at sufficient frame rates.

	Brain	Fullbody	Kidney
NVIDIA RTX 4070 TI Super	65	226	170
NVIDIA RTX A5000	68	341	199
AMD Ryzen™ 9 7900X iGPU	12	42	16

Table 3: Rendering performance at 2048x2048 resolution in frames per second, averaged over all training images and presets. For the iGPU a resolution of 1024x1024 is used.

6 Discussion and Outlook

Our evaluations show that compressed 3DGS enables interactive Cinematic Anatomy with datasets so large that this would have been impossible. This is possible by restricting to static presets. We are confident that this limitation is acceptable for educational use since not more than a few transfer function settings are usually selected. Since the memory requirements of compressed 3DGS are so low, a separate Gaussian representation can be computed for each preset.

In the images we have used so far to perform a qualitative comparison between compressed 3DGS and path tracing, static lighting conditions have been simulated with an environment map that does not change relative to the object. Thus, the object points are seen under the same lighting condition in every view, resulting in rather smooth illumination when changing the perspective under which the viewer looks at the object. This, however, changes when a headlight is used, and a point's illumination varies with varying camera position. The use of a headlight is demonstrated in Fig. 11. Notably, while most regions can be resolved very well by 3DGS, in some other regions the novel views show reconstruction artifacts. The strong variation of the reflected light under an illumination that changes in every image cannot be captured well by the 3D Gaussian representation. One approach we see to address this limitation is via re-lighting. By generating training images with optical material properties instead of illumination, it might be possible to better recover highly varying lighting conditions at runtime.

An important component in volume rendering applications is an interactively moveable clip plane. Since 3DGS can only handle static scenes, previously unseen objects that become visible due to applying a clip plane (or points that are clipped away and reveal other points) cannot be recovered. Current extensions of 3DGS to 4DGS [57, 57] do not seem applicable in our scenario since they assume that only the positions of visible points change, but not the number of points. Another possibility is to allow only changes of the clip plane in discrete steps, and to compute a separate Gaussian representation for each step. We are confident that a fairly compact representation can be obtained by progressively encoding the object points that appear and disappear when making subsequent steps.

As we have demonstrated, highly transparent volumes cannot be handled very well by 3DGS. Another example demonstrating this is shown in Fig. 12. As long as the volume is homogeneous and doesn't contain too many interior structures, a quite accurate reconstruction can be achieved. However, we have observed that with increasing depth complexity it becomes more and more difficult for 3DGS to represent all possible color and opacity distributions with a reasonable number of 3D Gaussians. A possible strategy to address this limitation might be to augment 3DGS with 3D Gaussians that are optimized with respect to an object-space loss, similar in spirit to EWA volume splatting [61].

7 Conclusion

In this work we have demonstrated the use of differentiable 3D Gaussian splatting for novel view synthesis from path traced images of high resolution medical datasets. We have shown that the 3D Gaussian representation can be compressed — at hardly perceivable loss in image quality — to a size that enables download and storage on even mobile devices. Even though the Gaussian representation needs to be re-generated for every selected preset, even when many different presets are used the overall memory is still far below the memory required by the dataset. Computationally expensive path or ray tracing can be avoided at rendering time, enabling fast display on mid- and even low-end devices.

We have also pointed at current limitations of 3DGS in the envisioned scenario. As the most crucial ones we see the current absence of support for clip planes and the quality degradation when semi-transparent volume rendering is performed. We have sketched future research directions to address these limitations, and we are confident that significant improvements in such scenarios can be achieved. Another challenging aspect that needs to be addressed in the future is the handling of time-varying dataset. We see more and more scanning technologies that can accurately measure blood flow and deforming tissue. Tailoring 3DGS for interactively monitoring and inspecting such dynamic processes is another important goal.

Acknowledgments

HiP-CT data from the UCL led European Synchrotron Radiation Facility (ESRF) beamtime MD1290 (courtesy Lee, Walsh, Tafforeau et al.), Nature Methods, Walsh et al 2021 [51].

References

[1] A. Adinets and D. Merrill. Onesweep: A Faster Least Significant Digit Radix Sort for GPUs. *arXiv preprint arXiv:2206.01784*, 2022. 6

[2] I. Baseer, I. Valverde, A. H. Moustafa, J. Blat, and O. Camara. Evaluation of cinematic volume rendering open-source and commercial solutions for the exploration of congenital heart data. In *2023 IEEE Visualization and Visual Analytics (VIS)*, pp. 76–80, 2023. doi: 10.1109/VIS54172.2023.00024 3

[3] J. Binder, C. Krautz, K. Engel, R. Grützmann, F. A. Fellner, P. H. Burger, and M. Scholz. Leveraging medical imaging for medical education — a

cinematic rendering-featured lecture. *Annals of Anatomy - Anatomischer Anzeiger*, 222:159–165, 2019. doi: 10.1016/j.aanat.2018.12.004 2

[4] J. S. Binder, M. Scholz, S. Ellmann, M. Uder, R. Grützmann, G. F. Weber, and C. Krautz. Cinematic rendering in anatomy: A crossover study comparing a novel 3d reconstruction technique to conventional computed tomography. *Anatomical Sciences Education*, 14(1):22–31, 2021. doi: 10.1002/ase.1989 2

[5] C. Birklbauer and O. Bimber. Light-field supported fast volume rendering. In *ACM SIGGRAPH 2012 Posters*, SIGGRAPH '12, article no. 125, 1 pages. Association for Computing Machinery, New York, NY, USA, 2012. doi: 10.1145/2342896.2343040 2

[6] U. D. Bordoloi and H.-W. Shen. View selection for volume rendering. In *VIS 05. IEEE Visualization, 2005.*, pp. 487–494. IEEE, 2005. 3

[7] E. Brochu, V. M. Cora, and N. de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *CoRR*, abs/1012.2599, 2010. 3

[8] S. Bruton, D. Ganter, and M. Manzke. Fast approximate light field volume rendering: Using volume data to improve light field synthesis via convolutional neural networks. In A. P. Cláudio, K. Bouatouch, M. Chessa, A. Paljic, A. Kerren, C. Hurter, A. Tremeau, and G. M. Farinella, eds., *Computer Vision, Imaging and Computer Graphics Theory and Applications*, pp. 338–361. Springer International Publishing, Cham, 2020. 2

[9] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[10] M. Chen and H. Jäenicke. An information-theoretic framework for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1206–1215, 2010. 3

[11] Z. Chen and H. Zhang. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5939–5948, 2019. 2

[12] D. Comaniciu, K. Engel, B. Georgescu, and T. Mansi. Shaping the future through innovations: From medical imaging to precision medicine. *Medical Image Analysis*, 33:19–26, 2016. 20th anniversary of the Medical Image Analysis journal (MedIA). doi: 10.1016/j.media.2016.06.016 1, 3

[13] E. Dappa, K. Higashigaito, J. Fornaro, S. Leschka, S. Wildermuth, and H. Alkadhi. Cinematic rendering – an alternative to volume rendering for 3d computed tomography imaging. *Insights into Imaging*, 7, 09 2016. doi: 10.1007/s13244-016-0518-1 3

[14] F. A. Fellner, K. Engel, and C. Kremer. Virtual anatomy: The dissecting theatre of the future—implementation of cinematic rendering in a large 8 k high-resolution projection environment. *Journal of Biomedical Science and Engineering*, 10:367–375, 2017. 2

[15] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa. Plenoxels: Radiance fields without neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5501–5510, June 2022. 2

[16] R. Garnett. *Bayesian Optimization*. Cambridge University Press, 2023. 3

[17] P. A. Glemser, K. Engel, D. Simons, J. Steffens, H.-P. Schlemmer, and B. Orakcioglu. A new approach for photorealistic visualization of rendered computed tomography images. *World Neurosurgery*, 114:e283–e292, 2018. doi: 10.1016/j.wneu.2018.02.174 2

[18] S. J. Gortler, R. Grzeszczuk, R. Szeliski, and M. F. Cohen. *The Lumigraph*. Association for Computing Machinery, New York, NY, USA, 1 ed., 2023. 2

[19] L. G. Henyey and J. L. Greenstein. Diffuse radiation in the galaxy. *Astrophysical Journal*, 93:70–83, 01 1941. doi: 10.1086/144246 3

[20] N. Hofmann, J. Martschinke, K. Engel, and M. Stamminger. Neural denoising for path tracing of medical volumetric data. *Proc. ACM Comput. Graph. Interact. Tech.*, 3(2), article no. 13, 18 pages, aug 2020. doi: 10.1145/3406181 2

[21] J. A. Iglesias-Guitian, P. Mane, and B. Moon. Real-time denoising of volumetric path tracing for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 28(7):2734–2747, 2022. doi: 10.1109/TVCG.2020.3037680 2

[22] S. Jabbireddy, S. Li, X. Meng, J. E. Terrill, and A. Varshney. Accelerated Volume Rendering with Volume Guided Neural Denoising. In T. Hoellt, W. Aigner, and B. Wang, eds., *EuroVis 2023 - Short Papers*. The Eurographics Association, 2023. doi: 10.2312/evs.20231042 2

[23] G. Ji and H.-W. Shen. Dynamic view selection for time-varying volumes. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1109–1116, 2006. 3

[24] X. Jiayi, T. Gaspard, C. Timothee, M. Matthew, L. Forrest, B. Tom, M. Ken, L. Yueh, and A. Stephen. Interactive, in-browser cinematic volume rendering of medical images. *Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization*, 11(4):1019–1026, 2023. doi: 10.1080/21681163.2022.2145239 3

[25] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. 2, 4, 5

[26] T. Kroes, F. Post, and C. Botha. Exposure render: An interactive photo-realistic volume rendering framework. *PloS one*, 7:e38586, 07 2012. doi: 10.1371/journal.pone.0038586 3

[27] J. C. Lee, D. Rho, X. Sun, J. H. Ko, and E. Park. Compact 3d gaussian representation for radiance field. In *CVPR*, 2024. 2

[28] M. Levoy and P. Hanrahan. *Light Field Rendering*. Association for Computing Machinery, New York, NY, USA, 1 ed., 2023. 2

[29] L. Li, Z. Shen, Z. Wang, L. Shen, and L. Bo. Compressing volumetric radiance fields to 1 mb. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4222–4231, June 2023. 2

[30] Y. Lu, K. Jiang, J. A. Levine, and M. Berger. Compressive neural representations of volumetric scalar fields. *Computer Graphics Forum*, 40(3):135–146, 2021. 2

[31] N. Marsaglia, Y. Kawakami, S. D. Schwartz, S. Fields, and H. Childs. An entropy-based approach for identifying user-preferred camera positions. In *2021 IEEE 11th Symposium on Large Data Analysis and Visualization (LDAV)*, pp. 73–83. IEEE Computer Society, Los Alamitos, CA, USA, oct 2021. doi: 10.1109/LDAV53230.2021.00015 3

[32] N. Marsaglia, M. Mathai, S. Fields, and H. Childs. Automatic in situ camera placement for isosurfaces of large-scale scientific simulations. In *EGPGV@ EuroVis*, pp. 49–59, 2022. 3

[33] J. Martschinke, S. Hartnagel, B. Keinert, K. Engel, and M. Stamminger. Adaptive temporal sampling for volumetric path tracing of medical data. *Computer Graphics Forum*, 38(4):67–76, 2019. doi: 10.1111/cgf.13771 2

[34] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4460–4470, 2019. 2

[35] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision – ECCV 2020*, pp. 405–421, 2020. doi: 10.1007/978-3-030-58452-8_24 1

[36] J. Mockus. *Bayesian Approach to Global Optimization: Theory and Applications*. Springer Netherlands, Dordrecht, 1989. doi: 10.1007/978-94-009-0909-0 3

[37] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, article no. 102, 15 pages, July 2022. doi: 10.1145/3528223.3530127 2

[38] S. Niedermayr, J. Stumpfegger, and R. Westermann. Compressed 3d gaussian splatting for accelerated novel view synthesis, 2024. 1, 2, 4, 5, 6

[39] F. Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014–. 3

[40] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 165–174, 2019. 2

[41] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In B. Leibe, J. Matas, N. Sebe, and M. Welling, eds., *Computer Vision – ECCV 2016*, pp. 525–542. Springer International Publishing, Cham, 2016. 4

[42] D. Rho, B. Lee, S. Nam, J. C. Lee, J. H. Ko, and E. Park. Masked wavelet representation for compact neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20680–20690, June 2023. 2

[43] T. Steffen, S. Winklhofer, F. Starz, D. Wiedemeier, U. Ahmadli, and B. Stadlinger. Three-dimensional perception of cinematic rendering versus conventional volume rendering using ct and cbct data of the facial skeleton. *Annals of Anatomy - Anatomischer Anzeiger*, 241:151905, 2022. doi: 10.1016/j.aanat.2022.151905 2

[44] C. Sun, M. Sun, and H. Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *CVPR*, 2022. 2

[45] S. Takahashi, I. Fujishiro, Y. Takeshima, and T. Nishita. A feature-driven approach to locating optimal viewpoints for volume visualization. In *VIS*

05. IEEE Visualization, 2005., pp. 495–502. IEEE, 2005. 3

[46] Y. Tao, H. Lin, H. Bao, F. Dong, and G. Clapworthy. Structure-aware viewpoint selection for volume visualization. In *2009 IEEE Pacific Visualization Symposium*, pp. 193–200. IEEE, 2009. 3

[47] Y. Tao, Q. Wang, W. Chen, Y. Wu, and H. Lin. Similarity voting based viewpoint selection for volumes. In *Computer graphics forum*, vol. 35, pp. 391–400. Wiley Online Library, 2016. 3

[48] P.-P. Vázquez, E. Monclús, and I. Navazo. Representative views and paths for volume models. In *International Symposium on Smart Graphics*, pp. 106–117. Springer, 2008. 3

[49] D. Vlasic, H. Pfister, S. Molinov, R. Grzeszczuk, and W. Matusik. Opacity light fields: interactive rendering of surface light fields with view-dependent opacity. In *Proceedings of the 2003 Symposium on Interactive 3D Graphics*, I3D '03, 10 pages, p. 65–74. Association for Computing Machinery, New York, NY, USA, 2003. doi: 10.1145/641480.641496 2

[50] C. Walsh, P. Tafforeau, W. Wagner, D. Jafree, A. Bellier, C. Werlein, M. Kühnel, E. Boller, S. Walker-Samuel, J. Robertus, et al. Imaging intact human organs with local resolution of cellular structures using hierarchical phase-contrast tomography. *Nature methods*, 18(12):1532–1541, 2021. 1

[51] C. Walsh, P. Tafforeau, W. Wagner, D. Jafree, A. Bellier, C. Werlein, M. Kühnel, E. Boller, S. Walker-Samuel, J. Robertus, D. Long, J. Jacob, S. Marussi, E. Brown, N. Holroyd, D. Jonigk, M. Ackermann, and P. Lee. Imaging intact human organs with local resolution of cellular structures using hierarchical phase-contrast tomography. *Nature Methods*, 18, 11 2021. doi: 10.1038/s41592-021-01317-x 5, 9

[52] J. Wasserthal. Dataset with segmentations of 117 important anatomical structures in 1228 ct images, oct 2023. doi: 10.5281/zenodo.10047292 5

[53] S. Weiss, P. Hermüller, and R. Westermann. Fast neural representations for direct volume rendering. *arXiv preprint*, 2021. doi: 10.48550/arXiv.2112.01579 2

[54] S. Weiss, P. Hermüller, and R. Westermann. Fast neural representations for direct volume rendering. *Computer Graphics Forum*, 41(6):196–211, 2022. doi: 10.1111/cgf.14578 3

[55] S. Weiss and R. Westermann. Differentiable direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 28(1):562–572, 2022. doi: 10.1109/TVCG.2021.3114769 2

[56] E. R. Woodcock, T. Murphy, P. J. Hemmings, and T. C. Longworth. Techniques used in the GEM code for Monte Carlo neutronics calculations in reactors and other systems of complex geometry. *Applications of Computing Methods to Reactor Problems*, 1965. 3

[57] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and W. Xinggang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023. 9

[58] C. Yang, Y. Li, C. Liu, and X. Yuan. Deep learning-based viewpoint recommendation in volume visualization. *Journal of Visualization*, 22(5):991–1003, 2019. 3

[59] Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger. Mip-splatting: Alias-free 3d gaussian splatting. *arXiv:2311.16493*, 2023. 2, 5

[60] Y. Yuan, J. Yang, Q. Sun, Y. Huang, and S. Ma. Cinematic volume rendering algorithm based on multiple lights photon mapping. *Multimedia Tools and Applications*, 83:1–14, 05 2023. doi: 10.1007/s11042-023-15075-9 2

[61] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross. EWA volume splatting. In *Proceedings Visualization, 2001. VIS '01.*, pp. 29–538. IEEE, San Diego, CA, USA, 2001. doi: 10.1109/VISUAL.2001.964490 1, 2, 4, 9

[62] U. Šmajdek, u. Lesar, M. Marolt, and C. Bohak. Combined volume and surface rendering with global illumination caching. *The Visual Computer*, pp. 1–13, 06 2023. doi: 10.1007/s00371-023-02932-9 2