Vision, Modeling, and Visualization (2024)
L. Linsen and J. Thies (Eds.)

Application of 3D Gaussian Splatting for Cinematic Anatomy on Consumer Class Devices

S. Niedermayr[1], C. Neuhauser[1], K. Petkov[2], K. Engel[2], R. Westermann[1]

[1]Technical University of Munich, [2]Siemens Healthineers

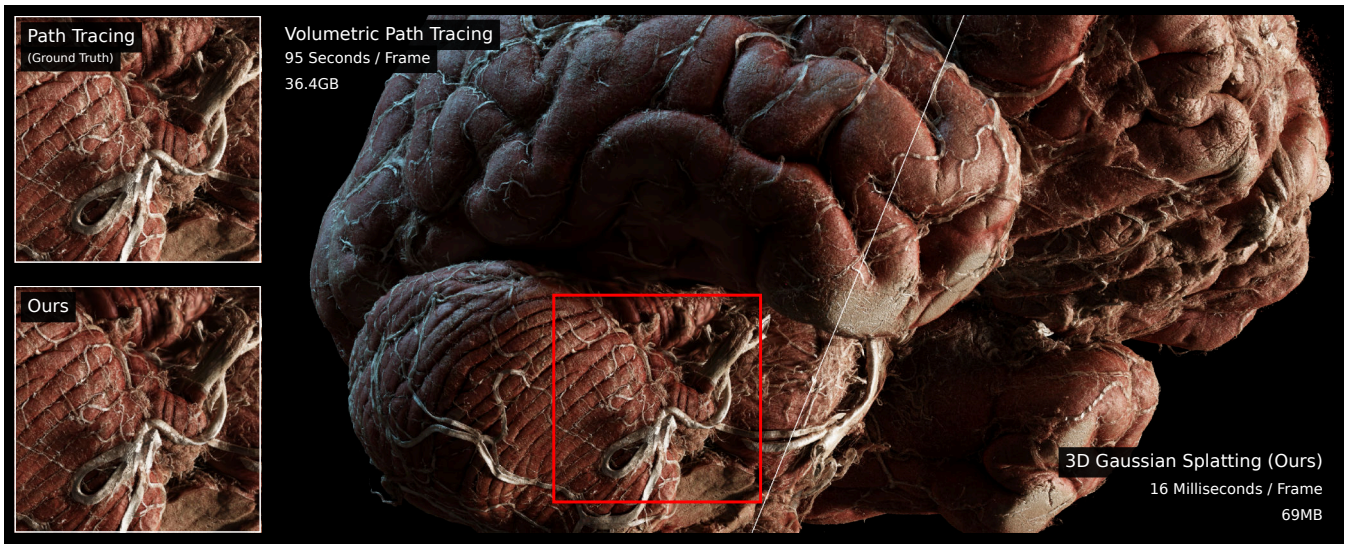


Figure 1: *Left: Path-traced image of a 36 GB HiP-CT data set, rendered in 95 seconds on a high-end GPU. Right: Gaussian splat representation of the same data set requiring 69 MB and rendered at 60 frames per second. The pixel resolution is 2048x2048.*

Abstract

Interactive photorealistic rendering of 3D anatomy is used in medical education to explain the structure of the human body. It is currently restricted to frontal teaching scenarios, where even with a powerful GPU and high-speed access to a large storage device where the data set is hosted, interactive demonstrations can hardly be achieved. We present the use of novel view synthesis via compressed 3D Gaussian Splatting (3DGS) to overcome this restriction, and to even enable students to perform cinematic anatomy on lightweight and mobile devices. Our proposed pipeline first finds a set of camera poses that captures all potentially seen structures in the data. High-quality images are then generated with path tracing and converted into a compact 3DGS representation, consuming < 70 MB even for data sets of multiple GBs. This allows for real-time photorealistic novel view synthesis that recovers structures up to the voxel resolution and is almost indistinguishable from the path-traced images.

CCS Concepts

• ***Computing methodologies*** → *Computer graphics; Rendering;*

1. Introduction

Cinematic Anatomy (CA) is an immersive anatomy learning application, which is designed to improve anatomy education through the use of photorealistic 3D rendering via path-tracing [CEGM16]. Instead of real 3D anatomy models, it utilizes volume data provided by medical scanning devices. The application is used in the field of anatomy education, e.g., in the JKU medSPACE [JKU], a lecture space for teaching anatomy. It is used for teaching the diverse and complex individual human anatomy, anatomical variations, and pathology, to enhance learners' competency with im-

DOI: 10.2312/vmv.20241195

mersive photorealistic 3D visualization of data from real patients. Several studies have shown the benefits of using photo-realistic volumetric rendering of clinical volume data for teaching and understanding anatomy [GES*18, BKE*19, SWS*22].

Additionally to stereoscopic projection modes for frontal education, students need to run CA on their mobile devices for personalized learning experiences. In addition to significant performance losses on such devices, 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 [WTW*21]. Thus, CA 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 data set is stored. As Fig. 1 demonstrates, even then is it difficult to render the data at interactive rates.

We demonstrate that differentiable 3DGS [KKLD23], which reconstructs a 3D Gaussian scene representation from images of this scene, can address the limitations of CA. The Gaussian representation can be rendered at high speed from arbitrary views, avoiding time-consuming path tracing. In combination with compressed 3DGS [NSW24], the memory consumption of the Gaussian representation is significantly reduced, and with GPU rasterization, 3D Gaussian splatting runs efficiently even on mobile devices. Fig. 1 demonstrates these properties with a high-resolution CT scan.

Contribution. To use compressed 3DGS for CA, we propose a processing pipeline including the following adaptations:

- We extend the view selection method proposed by Kopanas and Drettakis [KD23] for volume rendering to automatically find a set of cameras that captures all potentially seen structures under the current transfer function setting.
- We extend 3DGS with differentiable alpha channel rendering to create background-free reconstructions and drastically improve the reconstruction of translucent materials.
- Our implementation is open-source and available on GitHub under the link `https://github.com/KeKsBoTer/cinematic-gaussians`.

We analyze the quality, performance, and memory requirements with several high-resolution data sets. Training images are rendered with a publically available CA tool. The results demonstrate that the memory requirement is significantly below the initial data size. Since the renderable representation is so small, students can quickly download it over low-bandwidth channels and render on their mobile devices. Rendering performance is about two orders of magnitudes faster than optimized path tracing, with almost no perceptible loss of image quality.

Limitations. The use of 3DGS for CA comes with the following limitations: Firstly, lighting conditions are baked into the 3D Gaussian representation and cannot be changed during rendering. Secondly, due to the use of preset transfer functions and clip planes, the approach is less effective in supporting interactive volume exploration. Overcoming these limitations is difficult, and we discuss possible improvement strategies at the end of our work.

2. Related Work

3DGS [KKLD23] builds upon elliptical weighted average (EWA) volume splatting [ZPVBG01] 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 [YCH*23] 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 [NSW24, LRS*24]. For typical scenes, the memory requirements of 3DGS is below 50 MB without any noticeable differences in the reconstructed images.

3DGS for novel view synthesis overcomes in particular the difficulties of voxel-based approaches [MST*20, FKYT*22] to deal with sparsity. Even though adaptive hash grids [MESK22], tensor decomposition [CXG*22] or variants using dedicated compression schemes [LSW*23, RLN*23] can effectively reduce the required memory, they use volume ray-casting and, thus, require high-end GPUs to achieve reasonable rendering performance. The same limitation holds for differentiable volume rendering [WW22], which, similar in spirit to 3DGS, optimizes optical properties on a dense voxel grid using image-based loss functions.

3DGS optimizes a 3D scene representation from photorealistic images of that scene, which are generated with volumetric path-tracing [PJH23, NSJ14, NGHJ18, NDSRJ20]. A number of approaches have previously attempted to improve the performance of path tracing via image denoising [HMES20, JLM*23, IGMM22], photon mapping [YYS*23], illumination caching [vLMB23] and adaptive temporal sampling [MHK*19]. Despite achieving remarkable performance gains at high quality, all these approaches require a rendering system with enormous memory resources to host high-resolution data sets as well as huge computational power to perform ray tracing with such data. It is fair to say that high-quality path-tracing on consumer class hardware is impossible today.

In principle, the memory requirements of CA can be addressed with Scene Representation Networks (SRNs) [MON*19, CZ19, PFS*19], i.e., fully-connected neural networks that learn to encode a surface model as an implicit 3D function. Lu *et al.* [LJLB21] demonstrate the use of SRNs for volume data compression, by overfitting a network to a volume data set. This approach, however, comes at the expense of subsequent network evaluations during rendering, which makes even GPU-friendly ray-marching implementations [WHW22] significantly slower than 3DGS.

A challenging problem in novel view synthesis is to find an as small as possible set of camera poses for generating the training and test images. Note that this problem is different to the problem of viewpoint optimization in visualization, where visualization parameters are optimized to find a single best viewpoint, e.g., by using entropy-based [JS06, VMN08, TLB*09, CJ10, WHW22] or similarity-based [TWC*16, YLLY19] loss functions that guide an optimizer. For SRNs, Kopanas and Drettakis [KD23] have introduced an algorithm to automatically optimize the placement of cameras so that improved coverage of a scene is achieved. We adapt this algorithm to work with volumetric data sets.

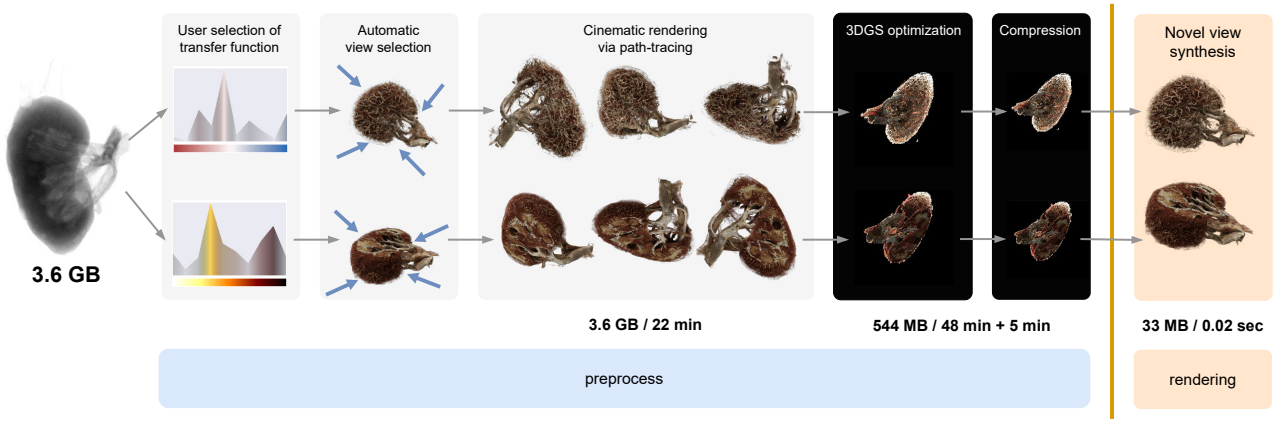


Figure 2: *CA pipeline for a* $1510 \times 1706 \times 1415$ *HiP-CT data set requiring 3.6 GB of memory. Numbers below each stage indicate the required memory at this stage and its computation times. 3D Gaussian splatting (3DGS) optimization uses 99 path traced images, and first generates the raw Gaussian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

3. Cinematic Anatomy Pipeline

The different stages of the proposed CA pipeline are shown in Fig. 2. After loading a data set, 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, multiple views capturing all potentially seen structures in the data are computed (cf. Section 3.1). In this way, we recover structures in the final object representation which are not seen when generating images with camera positions on a surrounding sphere. 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. Section 3.2).

Once the images for a selected preset have been rendered, 3DGS generates a set of 3D Gaussian splats with shape and appearance attributes so that their rendering matches the given images. Once the parameters of the Gaussians are computed via differentiable rendering, they are compressed using sensitivity-aware vector quantization and entropy encoding (cf. Section 3.3). 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. We embed Mip-Splatting [YCH*23] to account for different levels of detail and enable smooth transitions when the focal length is increased.

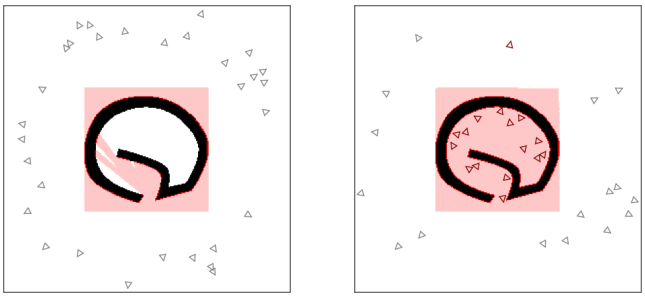
3.1. View Selection

Novel view synthesis requires that all visible scene parts are covered in the training images. Kopanas and Drettakis [KD23] propose an automatic camera placement algorithm for SRNs which maximises observation frequency and angular uniformity. The observation frequency lies between 0 (no camera observes a point) and 1 (all cameras observe a point). The angular uniformity considers the total variation distance between a 2D histogram of the directions of cameras observing a point in spherical coordinates and a uniform distribution. The algorithm iterates over batches of 1000 randomly sampled camera poses. In each iteration the cameras lying inside of or too close to occupied space are rejected. From the remaining camera poses the one is selected resulting in the highest improvement of the reconstruction, measured by observation frequency and angular uniformity.

We propose two modifications of this algorithm for 3DGS of volumetric data sets. Firstly, we observe that due to the high dimensionality of the search space, a huge number of cameras needs to be evaluated and optimal camera poses might be missed. To avoid this, we use Bayesian Optimal Sampling (BOS) [Moc89, Gar23] to adaptively place cameras in regions that are more promising to yield an improved maximum (called *exploitation*) or in previously rather unexplored regions (called *exploration*). In this way, the chance of missing optimal camera poses is significantly reduced, and fewer training images need to be generated. Secondly, instead of using a binary visibility indicating whether a point lies inside or outside the camera frustum, we use a continuous visibility that also considers (partial) occlusion. At each voxel, GPU ray marching is performed to compute the maximum transmittance, which is then used as a visibility indicator. For data sets not fitting into GPU memory, we perform this step with a lower-resolution copy.

BOS applies a probabilistic (usually Gaussian) surrogate model and an acquisition function. The former expresses Bayesian belief about the output of the objective function derived from prior evaluations, and the latter is used for selecting the next set of parameters for evaluating the objective function. For the acquisition function, the upper confidence bound [BCdF10] is used with parameter $\kappa = 10$, which controls the trade-off between exploitation and exploration. The value was empirically determined to work well with all test data sets, but can also be subjected to hyperparameter optimization either regarding the energy term by Kopanas and Drettakis [KD23] or the reconstruction quality with respect to images in a training set. We make use of the publically available software library Limbo [CCAM18] to perform the optimization process.

In Fig. 3, we showcase the ability of automatic view selection using BOS for improved capturing of internal structures of a concave



(a) *Pure hemisphere sampling.* **(b)** *Bayesian optimal sampling.*

Figure 3: *View selection strategies for a concave 2D test data set (black). Left: Random sampling of camera poses on a bounding circle cannot capture interior structures. Right: With BOS, given a sparse set of initial poses (grey), additional poses (red) are automatically determined to capture otherwise not seen object parts. Red area indicates seen domain parts.*

2D test data set. The view selection algorithm has been adapted to 2D for illustrative purposes. When randomly sampling cameras on the hemisphere around an object, structures in the interior are missed. When sampling only a small set of cameras on the hemisphere, BOS selects few additional camera poses which capture insufficiently seen structures on the outside and not yet captured structures on the inside. We demonstrate the resulting improvements in the reconstruction quality with 3DGS in Section 4.3. We further compare the convergence rate and performance of BOS and random sampling [KD23] in the supplementary material.

3.2. Image Generation

We render volume data with Monte Carlo volume path tracing from multiple views to generate a set of training images. Delta tracking [WMHL65] is used to determine a scattering event (the Henyey-Greenstein phase-function [HG41] is applied to determine the next ray direction), an absorption event (the path is terminated and the emissive color is regarded as the path contribution) or a null collision (the path is followed unchanged). A surface intersection is assumed when the density gradient magnitude exceeds a user-specified iso-value. Global illumination is then simulated by generating a reflection event for the surface with the new ray direction sampled proportional to the probability density function of a chosen reflectance distribution function. This process is repeated until the ray leaves the volume domain or an absorption event happens.

High dynamic range light maps are employed to look up lighting information from the environment. 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 (based on the transfer function preset) and memory coherent scattering. The latter optimization ensures that rays of neighboring pixels are scattered in the same direction, thus ensuring optimized cache utilization.

3.3. Compressed Differentiable 3D Gaussian Splatting

Differentiable 3DGS describes an object by a set of 3D Gaussians

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

Each Gaussian is centered at $x \in \mathbb{R}^3$, and the covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$ describes 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.

The 2D projection of a 3D Gaussian is a 2D Gaussian with a covariance that is derived from the view transformation matrix and the Jacobian of the affine approximation of the projective transformation. The scene is rendered by projecting all Gaussian into the image plane in sorted order and blending their contributions.

While Zwicker *et al.* [ZPVBG01] model a 3D scalar field via a set of 3D Gaussians so that the field can be reconstructed sufficiently well, Kerbl *et al.* [KKLD23] 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. The optimization process removes some of the initially selected 3D Gaussians (if no contribution), adaptively splits Gaussians, and modifies their shapes and appearance attributes to minimize an image-based loss function.

To further reduce the memory consumption of 3DGS, we utilize the compression proposed by Niedermayr *et al.* [NSW24]. It encodes SH coefficients and Gaussian shape parameters into compact codebooks via sensitivity-aware vector quantization, and then fine-tunes the parameters on the training images. Quantization-aware training [RORF16] is used to represent the scene parameters with fewer bits. We call this strategy High-Rate-compression (HR-compression). We also provide an option that uses only quantization-aware training to reduce all scene parameters but the Gaussians' positions to an 8-bit representation during optimization. We will subsequently call this strategy High-Quality-compression (HQ-compression). Since CA requires the entire data set in focus, we can omit the scaling factor that is usually stored per Gaussian to represent scenes with objects in focus and surrounding background.

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. Therefore, we extend 3DGS to allow for differentiable rendering of images with alpha channel. We use a combination of per pixel L1 and SSIM Loss to accurately reconstruct the alpha channel of the volume rendered training images. As shown in Section 4.4, the reconstruction quality can be improved significantly in this way.

4. Results and evaluation

We analyze the performance, memory consumption and quality of the proposed pipeline for CA with a variety of high-resolution medical data sets showing different anatomical structures. Our 3DGS

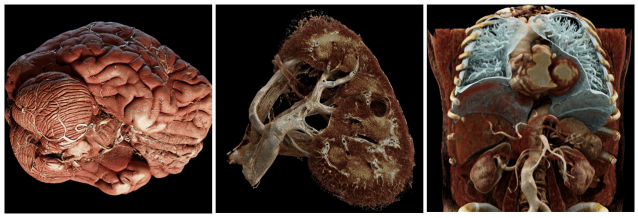


Figure 4: *From left to right, Brain (*$3224 \times 3224 \times 3585$*), Kidney (*$1510 \times 1706 \times 1415$*) and Body (*$317 \times 317 \times 835$*). All images rendered at full HD with HR-compressed 3D Gaussian splatting using* < 70 *MB per data set at* > 30 *frames per second.*

implementation is a modification of the code provided by Kerbl *et al*. [KKLD23]. For compression and rendering, we use the settings described by Niedermayr *et al*. [NSW24].

4.1. Data Sets

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 [WTW*21].

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.

Body is a human CT angiography scan at resolution $317 \times 317 \times 835$ from the collection by Wasserthal [Was23], image id *s0287*. The data set contains some semi-transparent material showing significant differences under directional lighting. We render it under complex lighting conditions to challenge 3DGS's reconstruction capabilities.

We show all data sets in Fig. 4, and provide an interactive online demonstration at `https://keksboter.github.io/cinematic-gaussians/`. For each data set, between one and three presets have been used, including segmentations, transfer function and lighting conditions. 3DGS optimization has been performed on training images of resolution 2048×2048.

4.2. Preprocessing

With a GPU providing sufficient RAM, the initial images of all data sets can be generated with the publically available CA package, and using the built-in animation system to generate the views. We have used a research version providing batch rendering support, running on an NVIDIA A100 GPU for *Brain* and an NVIDIA RTX A5000 for *Kidney* and *Body*.

Table 1 shows in columns *Size* the size of each data set 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. 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.

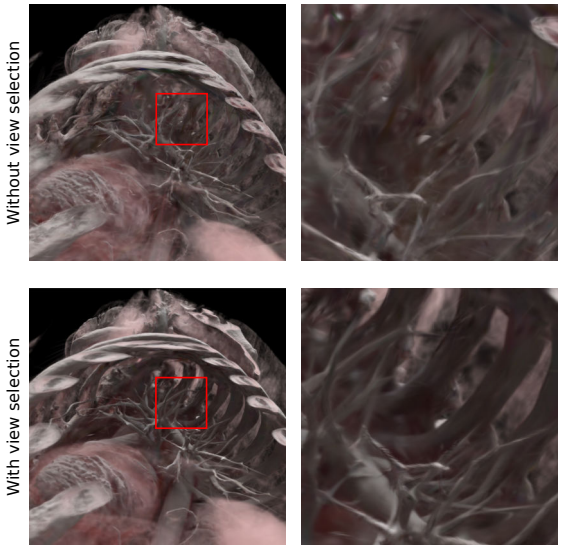


Figure 5: *Automatic view selection generates camera poses covering unseen parts of the volume, like the inside of the rib cage.*

4.3. View Selection

Automatic view selection is demonstrated with Body, which exhibits a lot of structures that 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. 5). As can be seen, overall improved reconstruction quality of parts not seen with random camera selection is achieved.

Table 1: *Memory requirements and preprocessing performance.*

	Path Tracing			3DGS (HR-Compression)		
	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
Body	0.2 GB	23 Min	99	7 MB	50 Min	0.9 M

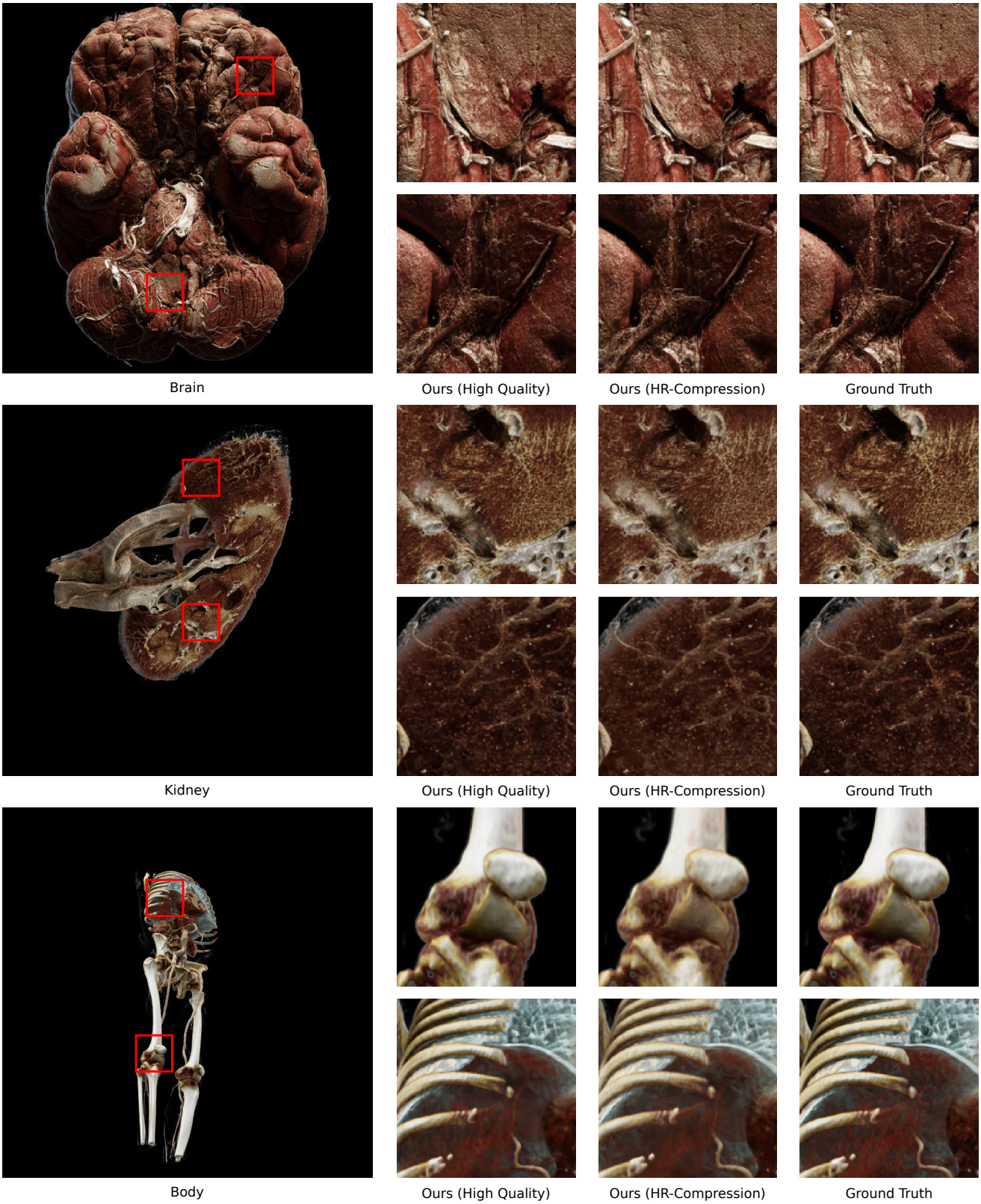


Figure 6: *Quality comparison for HQ-compressed and HR-compressed Gaussian representations. All images are from the test set.*

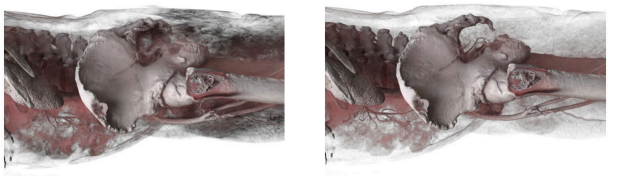


Figure 7: *Color-only reconstruction (left) leads to reconstruction artifacts, which disappear when 3DGS is optimized for color and opacity (right).*

4.4. Quality Evaluation

Fig. 6 compares test images that have not been seen during 3DGS optimization to images rendered with HQ- and HR-compressed 3DGS. Close-up views reveal only subtle color shifts between path traced images and images generated via HR-compressed 3DGS. HQ-compression leads to an increase in memory by a factor of three, yet differences in image quality are further reduced and become so small that they are hardly noticeable by eye.

Notably, significant losses in reconstruction quality are introduced when differentiable 3DGS optimizes only for RGB color (see Fig. 7 for an example). 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.

Table 2 shows the average SSIM and PSNR between the test images and the novel views rendered with HR-compressed 3DGS, averaged over all presets. 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 truths.

Table 2: *Quantitative evaluation of HR-compression,*

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

We further shed light on the capabilities of 3DGS to reconstruct semi-transparent regions in a data set. *Body* is used with a preset so that certain tissue types in the data set become semi-transparent, see Fig. 8. While overall the novel view matches the test images fairly well, the closeup views show that some fine details are not reconstructed accurately, and especially the semi-transparent structures are blurred out. This effect increases with increasing depth complexity, since it becomes more and more difficult for 3DGS to represent all possible color and opacity distributions accurately.

When using a preset with strong directional lighting from the environment map, one observes some high-frequency illumination variations especially in the volumetric regions. This makes it more difficult for 3DGS to accurately recover the tissue structures. Interestingly, Fig. 9 demonstrates that the reconstruction works very

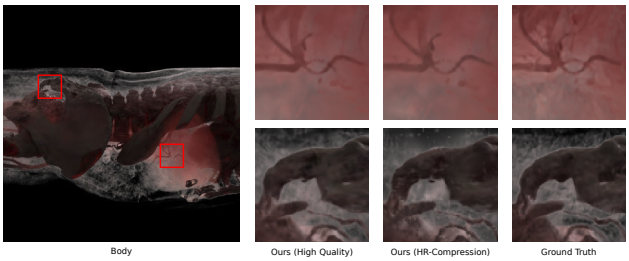


Figure 8: *Quality comparison for HQ-compressed and HR-compressed Gaussian representations of semi-transparent regions.*

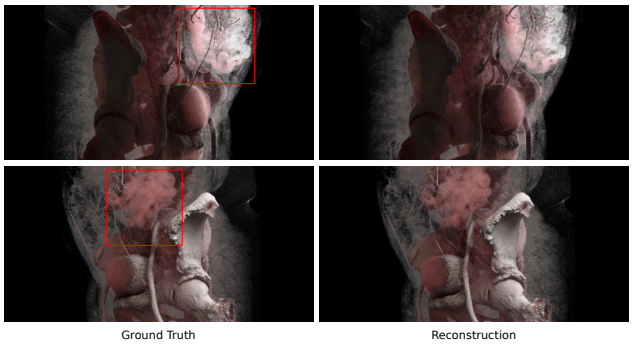


Figure 9: *Complex view-dependent lighting effects are well preserved by compressed 3DGS, even for semi-transparent material. The tissue marked with a red box shows high-frequent color variation under different perspectives.*

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 optimize a significantly increased number of parameters, requiring far more Gaussians to accurately represent the data.

4.5. Rendering Performance

The WebGPU implementation by Niedermayr *et al.* [NSW24] is used for performance testing. It enables rendering of compressed 3DGS up to 4× faster than the renderer by Kerbl *et al.* [KKLD23] and runs in a modern browser.

Table 3 shows that the rendering times even on an integrated iGPU is higher than 10 frames per second for the biggest data set *Brain*. On current mid- to high-end GPUs, 60 frames per second can be achieved for all data sets. This makes the CA pipeline especially 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. In

a supplementary video we demonstrate rendering performance of roughly 5 to 20 frames per second on a mobile device with Qualcomm Adreno 740 GPU.

	Brain	Body	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.*

5. Discussion and Outlook

Our experiments show that compressed 3DGS enables interactive CA with extremely large data sets, by restricting to static presets. We believe that this limitation is acceptable for educational use since not more than a few presets are usually selected. Since the memory requirement of compressed 3DGS is so low, a separate Gaussian representation can be computed for each preset.

In all experiments we have simulated static lighting conditions with an environment map that does not change relative to the object. Thus, the objects are seen under the same lighting condition in every view, resulting in rather smooth illumination when changing the camera pose. This, however, changes when a headlight is used, and a point's illumination varies with varying camera pose (see Fig. 10). Notably, while in this situation 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 3DGS. 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.

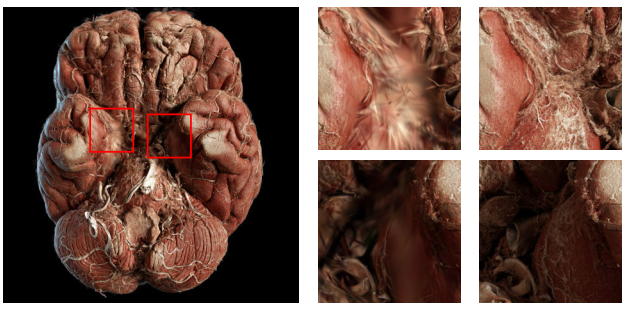


Figure 10: *With a headlight, 3DGS faces problems in some places to accurately reconstruct structures with high-frequent changes in illumination.*

A useful component for volume exploration is an interactive clip plane, resulting in object points that were previously unseen to become visible. One possibility to include clip planes is to restrict the plane movement to discrete steps and compute a separate Gaussian representation for each step. While this will significantly increase the memory requirements, we are confident that a fairly compact representation can be obtained by exploiting spatial coherence and progressively encoding the 3D Gaussians that appear and disappear when making subsequent steps.

6. Conclusion

We have demonstrated the use of differentiable 3DGS for novel view synthesis from path traced images of high resolution medical data sets. 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. The Gaussian representation needs to be re-generated for every selected preset, yet even for many presets the overall memory is still significantly below the memory required by the data set. Computationally expensive path 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 for CA. As the most important ones we see the current absence of support for clip planes and the quality degradation when a headlight is used. We have sketched future research directions to address these limitations, and we are confident that improvements can be achieved. There is also a pressing need to handle time-varying data sets, since we see more and more scanning technologies that can accurately measure blood flow and deforming tissue. Tailoring 3DGS for interactively visualizing such dynamic processes is another important goal.

Finally, we want to mention that besides CA we see in-situ visualization as another promising application of 3DGS. For data sets which are simulated on a supercomputer and are so large that they cannot be streamed out, images of the data set can be generated directly on the supercomputer and then streamed out to a system where novel view synthesis is performed. By using advanced implementations of 3DGS optimization, this might even become possible at rates enabling an explorative visual analysis.

References

[BCdF10] BROCHU E., CORA V. M., DE FREITAS N.: A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *CoRR* (2010). arXiv:1012.2599. 3

[BKE*19] BINDER J., KRAUTZ C., ENGEL K., GRÜTZMANN R., FELLNER F. A., BURGER P. H., SCHOLZ M.: Leveraging medical imaging for medical education — a cinematic rendering-featured lecture. *Annals of Anatomy - Anatomischer Anzeiger* (2019). doi:https://doi.org/10.1016/j.aanat.2018.12.004. 2

[CCAM18] CULLY A., CHATZILYGEROUDIS K., ALLOCATI F., MOURET J.-B.: Limbo: A Flexible High-performance Library for Gaussian Processes modeling and Data-Efficient Optimization. *The Journal of Open Source Software 3*, 26 (2018), 545. doi:10.21105/joss.00545. 3

[CEGM16] COMANICIU D., ENGEL K., GEORGESCU B., MANSI T.: Shaping the future through innovations: From medical imaging to precision medicine. *Medical Image Analysis* (2016). doi:https://doi.org/10.1016/j.media.2016.06.016. 1

[CJ10] CHEN M., JÄENICKE H.: An information-theoretic framework for visualization. *TVCG* (2010). 2

[CXG*22] CHEN A., XU Z., GEIGER A., YU J., SU H.: Tensorf: Tensorial radiance fields. In *ECCV* (2022). 2

[CZ19] CHEN Z., ZHANG H.: Learning implicit fields for generative shape modeling. In *CVPR* (2019). 2

[FKYT*22] FRIDOVICH-KEIL S., YU A., TANCIK M., CHEN Q., RECHT B., KANAZAWA A.: Plenoxels: Radiance fields without neural networks. In *CVPR* (June 2022). 2

[Gar23] GARNETT R.: *Bayesian Optimization*. Cambridge University Press, 2023. 3

[GES*18] GLEMSER P. A., ENGEL K., SIMONS D., STEFFENS J., SCHLEMMER H.-P., ORAKCIOGLU B.: A new approach for photorealistic visualization of rendered computed tomography images. *World Neurosurgery* (2018). doi:https://doi.org/10.1016/j.wneu.2018.02.174. 2

[HG41] HENYEY L. G., GREENSTEIN J. L.: Diffuse radiation in the galaxy. *Astrophysical Journal* (Jan. 1941). doi:10.1086/144246. 4

[HMES20] HOFMANN N., MARTSCHINKE J., ENGEL K., STAMMINGER M.: Neural denoising for path tracing of medical volumetric data. *ACM TOG* (Aug. 2020). doi:10.1145/3406181. 2

[IGMM22] IGLESIAS-GUITIAN J. A., MANE P., MOON B.: Real-time denoising of volumetric path tracing for direct volume rendering. *TVCG* (2022). doi:10.1109/TVCG.2020.3037680. 2

[JKU] JKU: JKU medSPACE. https://ars.electronica.art/futurelab/en/projects-jku-medspace/. Accessed: 2024-05-12. 1

[JLM*23] JABBIREDDY S., LI S., MENG X., TERRILL J. E., VARSHNEY A.: Accelerated Volume Rendering with Volume Guided Neural Denoising. In *EuroVis 2023 - Short Papers* (2023), Hoellt T., Aigner W., Wang B., (Eds.), The Eurographics Association. doi:10.2312/evs.20231042. 2

[JS06] JI G., SHEN H.-W.: Dynamic view selection for time-varying volumes. *TVCG* (2006). 2

[KD23] KOPANAS G., DRETTAKIS G.: Improving NeRF Quality by Progressive Camera Placement for Free-Viewpoint Navigation. In *VMV* (2023), The Eurographics Association. doi:10.2312/vmv.20231222. 2, 3, 4

[KKLD23] KERBL B., KOPANAS G., LEIMKÜHLER T., DRETTAKIS G.: 3d gaussian splatting for real-time radiance field rendering. *ACM TOG* (July 2023). 2, 4, 5, 7

[LJLB21] LU Y., JIANG K., LEVINE J. A., BERGER M.: Compressive neural representations of volumetric scalar fields. *CGF* (2021). 2

[LRS*24] LEE J. C., RHO D., SUN X., KO J. H., PARK E.: Compact 3d gaussian representation for radiance field. In *CVPR* (2024). 2

[LSW*23] LI L., SHEN Z., WANG Z., SHEN L., BO L.: Compressing volumetric radiance fields to 1 mb. In *CVPR* (June 2023). 2

[MESK22] MÜLLER T., EVANS A., SCHIED C., KELLER A.: Instant neural graphics primitives with a multiresolution hash encoding. *ACM TOG* (July 2022). doi:10.1145/3528223.3530127. 2

[MHK*19] MARTSCHINKE J., HARTNAGEL S., KEINERT B., ENGEL K., STAMMINGER M.: Adaptive temporal sampling for volumetric path tracing of medical data. *CGF* (2019). doi:https://doi.org/10.1111/cgf.13771. 2

[Moc89] MOCKUS J.: *Bayesian Approach to Global Optimization: Theory and Applications*. Springer Netherlands, 1989. doi:10.1007/978-94-009-0909-0. 3

[MON*19] MESCHEDER L., OECHSLE M., NIEMEYER M., NOWOZIN S., GEIGER A.: Occupancy networks: Learning 3d reconstruction in function space. In *CVPR* (2019). 2

[MST*20] MILDENHALL B., SRINIVASAN P. P., TANCIK M., BARRON J. T., RAMAMOORTHI R., NG R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In *Computer Vision – ECCV 2020* (2020). doi:10.1007/978-3-030-58452-8_24. 2

[NDSRJ20] NIMIER-DAVID M., SPEIERER S., RUIZ B., JAKOB W.: Radiative backpropagation: An adjoint method for lightning-fast differentiable rendering. *Transactions on Graphics (Proceedings of SIGGRAPH)* (2020). doi:10.1145/3386569.3392406. 2

[NGHJ18] NOVÁK J., GEORGIEV I., HANIKA J., JAROSZ W.: Monte carlo methods for volumetric light transport simulation. In *Computer graphics forum* (2018), vol. 37, Wiley Online Library, pp. 551–576. 2

[NSJ14] NOVÁK J., SELLE A., JAROSZ W.: Residual ratio tracking for estimating attenuation in participating media. *ACM Trans. Graph. 33*, 6 (2014), 179–1. 2

[NSW24] NIEDERMAYR S., STUMPFEGGER J., WESTERMANN R.: Compressed 3d gaussian splatting for accelerated novel view synthesis. In *CVPR* (2024). 2, 4, 5, 7

[PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR* (2019). 2

[PJH23] PHARR M., JAKOB W., HUMPHREYS G.: *Physically based rendering: From theory to implementation*. MIT Press, 2023. 2

[RLN*23] RHO D., LEE B., NAM S., LEE J. C., KO J. H., PARK E.: Masked wavelet representation for compact neural radiance fields. In *CVPR* (2023). 2

[RORF16] RASTEGARI M., ORDONEZ V., REDMON J., FARHADI A.: XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In *ECCV* (2016). 4

[SWS*22] STEFFEN T., WINKLHOFER S., STARZ F., WIEDEMEIER D., AHMADLI U., STADLINGER B.: Three-dimensional perception of cinematic rendering versus conventional volume rendering using ct and cbct data of the facial skeleton. *Annals of Anatomy - Anatomischer Anzeiger* (2022). doi:https://doi.org/10.1016/j.aanat.2022.151905. 2

[TLB*09] TAO Y., LIN H., BAO H., DONG F., CLAPWORTHY G.: Structure-aware viewpoint selection for volume visualization. In *2009 IEEE Pacific Visualization Symposium* (2009), IEEE. 2

[TWC*16] TAO Y., WANG Q., CHEN W., WU Y., LIN H.: Similarity voting based viewpoint selection for volumes. In *CGF* (2016), Wiley Online Library. 2

[vLMB23] ŠMAJDEK U., LESAR U., MAROLT M., BOHAK C.: Combined volume and surface rendering with global illumination caching. *The Visual Computer* (June 2023). doi:10.1007/s00371-023-02932-9. 2

[VMN08] VÁZQUEZ P.-P., MONCLÚS E., NAVAZO I.: Representative views and paths for volume models. In *International Symposium on Smart Graphics* (2008), Springer. 2

[Was23] WASSERTHAL J.: Dataset with segmentations of 117 important anatomical structures in 1228 ct images, oct 2023. doi:10.5281/zenodo.10047292. 5

[WHW22] WEISS S., HERMÜLLER P., WESTERMANN R.: Fast neural representations for direct volume rendering. *CGF* (2022). doi:https://doi.org/10.1111/cgf.14578. 2

[WMHL65] WOODCOCK E. R., MURPHY T., HEMMINGS P. J., LONGWORTH T. C.: 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). 4

[WTW*21] WALSH C., TAFFOREAU P., WAGNER W., JAFREE D., BELLIER A., WERLEIN C., KÜHNEL M., BOLLER E., WALKER-SAMUEL S., ROBERTUS J., LONG D., JACOB J., MARUSSI S., BROWN E., HOLROYD N., JONIGK D., ACKERMANN M., LEE P.: Imaging intact human organs with local resolution of cellular structures using hierarchical phase-contrast tomography. *Nature Methods* (Nov. 2021). doi:10.1038/s41592-021-01317-x. 2, 5

[WW22] WEISS S., WESTERMANN R.: Differentiable direct volume rendering. *TVCG* (2022). doi:10.1109/TVCG.2021.3114769. 2

[YCH*23] YU Z., CHEN A., HUANG B., SATTLER T., GEIGER A.: Mip-splatting: Alias-free 3d gaussian splatting. *CoRR* (2023). arXiv:2311.16493. 2, 3

[YLLY19] YANG C., LI Y., LIU C., YUAN X.: Deep learning-based viewpoint recommendation in volume visualization. *Journal of Visualization* (2019). 2

[YYS*23] YUAN Y., YANG J., SUN Q., HUANG Y., MA S.: Cinematic volume rendering algorithm based on multiple lights photon mapping. *Multimedia Tools and Applications* (May 2023). doi:10.1007/s11042-023-15075-9. 2

[ZPVBG01] ZWICKER M., PFISTER H., VAN BAAR J., GROSS M.: EWA volume splatting. In *VIS* (2001), IEEE. doi:10.1109/VISUAL.2001.964490. 2, 4