Exact-GS: Mathematically Rigorous and Accurate 3D Gaussian Splatting for 3D X-ray Reconstruction

Supplementary Material

8. Additional Method Details

8.1. CT Projection and Lambert-Beer's Law

In computed tomography (CT), attenuation refers to the reduction in X-ray beam energy as it passes through a material. This occurs due to physical processes like photoelectric absorption and Compton scattering. The degree of attenuation depends on the material's density ρ, the atomic weight of the material Z, and the X-ray photon energy E . The material-specific linear attenuation coefficient $\mu = \mu(\rho, Z, E)$ quantifies this property.

The Lambert-Beer Law mathematically describes the relationship between attenuation and intensity. I_0 is the X-ray beam's initial intensity from the source. Transmitted intensity I_s is the reduced intensity after passing through a distance s. For heterogeneous materials, the law generalizes to an integral form, where the total attenuation is the sum of contributions along the X-ray path:

$$I_s = I_0 \cdot \exp\left(-\int_0^s \mu(t)\, dt\right). \tag{14}$$

The CT projection along this path is defined for convenience as the negative logarithm of the ratio of I_s and I_0:

$$p(s) = -\ln\left(\frac{I_s}{I_0}\right) = \int_0^s \mu(t)\, dt. \tag{15}$$

It should be noted here that this continuous integration is more consistent with the physical properties of CT projection than voxel grid discrete integration in the real world.

8.2. Closed-Form Expression for $\boldsymbol{T}$

Fortunately, the Rotation matrix $\boldsymbol{T}(u, v, \mathrm{D})$ has a closed-form expression, We substitute all the variables

$$\boldsymbol{n} = \begin{bmatrix} \frac{v}{l\sin(\phi)} \\ \frac{-u}{l\sin(\phi)} \\ 0 \end{bmatrix}, \boldsymbol{N} = \begin{bmatrix} 0 & 0 & \frac{-u}{l\sin(\phi)} \\ 0 & 0 & \frac{-v}{l\sin(\phi)} \\ \frac{u}{l\sin(\phi)} & \frac{v}{l\sin(\phi)} & 0 \end{bmatrix}, \tag{16}$$

and

$$\cos(\phi) = \frac{\mathrm{D}}{l}, \quad l = \sqrt{u^2 + v^2 + \mathrm{D}^2}, \tag{17}$$

into the Eq. (9) and get the final form of the rotation matrix:

$$\boldsymbol{T}(u, v, \mathrm{D}) = \begin{bmatrix} 1 - n_2^2(1-\cos(\phi)) & n_1 n_2(1-\cos(\phi)) & n_2\sin(\phi) \\ n_1 n_2(1-\cos(\phi)) & 1 - n_1^2(1-\cos(\phi)) & -n_1\sin(\phi) \\ -n_2\sin(\phi) & n_1\sin(\phi) & 1 - (n_2^2+n_1^2)(1-\cos(\phi)) \end{bmatrix} \tag{18}$$

$$= \begin{bmatrix} 1 - \frac{u^2}{l(l+\mathrm{D})} & \frac{-uv}{l(l+\mathrm{D})} & -u/l \\ \frac{-uv}{l(l+\mathrm{D})} & 1 - \frac{v^2}{l(l+\mathrm{D})} & -v/l \\ u/l & v/l & 1 - \frac{u^2+v^2}{l(l+\mathrm{D})} \end{bmatrix} \tag{19}$$

$$= \begin{bmatrix} 1 - \frac{u^2}{\sqrt{u^2+v^2+\mathrm{D}^2}(\sqrt{u^2+v^2+\mathrm{D}^2}+\mathrm{D})} & \frac{-uv}{\sqrt{u^2+v^2+\mathrm{D}^2}(\sqrt{u^2+v^2+\mathrm{D}^2}+\mathrm{D})} & \frac{-u}{\sqrt{u^2+v^2+\mathrm{D}^2}} \\ \frac{-uv}{\sqrt{u^2+v^2+\mathrm{D}^2}(\sqrt{u^2+v^2+\mathrm{D}^2}+\mathrm{D})} & 1 - \frac{v^2}{\sqrt{u^2+v^2+\mathrm{D}^2}(\sqrt{u^2+v^2+\mathrm{D}^2}+\mathrm{D})} & \frac{-v}{\sqrt{u^2+v^2+\mathrm{D}^2}} \\ \frac{u}{\sqrt{u^2+v^2+\mathrm{D}^2}} & \frac{v}{\sqrt{u^2+v^2+\mathrm{D}^2}} & \frac{\mathrm{D}}{\sqrt{u^2+v^2+\mathrm{D}^2}} \end{bmatrix}. \tag{20}$$

8.3. Proof of the Exact Equality of Projection

Following the pixel space transformation, we can get the ray in pixel space:

$$\boldsymbol{r}_p(t) = \boldsymbol{T}\boldsymbol{r}_c = t\boldsymbol{T}\boldsymbol{d} \in \boldsymbol{R}^3. \tag{21}$$

Note that $\boldsymbol{T}$ is a rigid rotation matrix in Euclidean space, which belongs to an affine transform. Therefore, we can get the following relationship : $\boldsymbol{T}^T = \boldsymbol{T}^{-1}$. Consequently, the projection in pixel space is formulated as

$$\begin{aligned}
p_i(\boldsymbol{r}_p) &= \int_0^{+\infty} G_{i,p}(t\boldsymbol{T}\boldsymbol{d}|\rho_i, \boldsymbol{\mu}_{i,p}, \boldsymbol{\Sigma}_{i,p})\, dt \\
&= \int_0^{+\infty} \rho_i \cdot \exp\left(-\frac{1}{2}(t\boldsymbol{T}\boldsymbol{d} - \boldsymbol{T}\boldsymbol{\mu}_{i,c})^T(\boldsymbol{T}\boldsymbol{\Sigma}_{i,c}\boldsymbol{T}^\top)^{-1}(t\boldsymbol{T}\boldsymbol{d} - \boldsymbol{T}\boldsymbol{\mu}_{i,c})\right) dt \\
&= \int_0^{+\infty} \rho_i \cdot \exp\left(-\frac{1}{2}(t\boldsymbol{d} - \boldsymbol{\mu}_{i,c})^T\boldsymbol{T}^T\boldsymbol{T}\boldsymbol{\Sigma}_{i,c}^{-1}\boldsymbol{T}^\top\boldsymbol{T}(t\boldsymbol{d} - \boldsymbol{\mu}_{i,c})\right) dt \\
&= \int_0^{+\infty} \rho_i \cdot \exp\left(-\frac{1}{2}(t\boldsymbol{d} - \boldsymbol{\mu}_{i,c})^T\boldsymbol{\Sigma}_{i,c}^{-1}(t\boldsymbol{d} - \boldsymbol{\mu}_{i,c})\right) dt \\
&= p_i(\boldsymbol{r}_c).
\end{aligned} \tag{22}$$

Obviously the projection $p_i(\boldsymbol{r}_p)$ is equal to the projection in camera space $p_i(\boldsymbol{r}_c)$.

8.4. Derivation of Ray Tracing Rendering

The integral $p_i(\boldsymbol{r}_c)$ at pixel $\boldsymbol{x}_\mathrm{d}$ along $\boldsymbol{r}_c$ can also be exactly solved:

$$p_i(\boldsymbol{r}_c) = \rho_i \cdot \int_0^{+\infty} \exp\left(-\frac{1}{2}(t\boldsymbol{d} - \boldsymbol{\mu}_{i,c})^T\boldsymbol{\Sigma}_{i,c}^{-1}(t\boldsymbol{d} - \boldsymbol{\mu}_{i,c})\right) dt \tag{23}$$

$$= \rho_i \cdot \int_0^{+\infty} \exp\left(-\frac{1}{2}(t^2\boldsymbol{d}^\top\boldsymbol{\Sigma}_{i,c}^{-1}\boldsymbol{d} - 2t\boldsymbol{d}^\top\boldsymbol{\Sigma}_{i,c}^{-1}\boldsymbol{\mu}_{i,c} + \boldsymbol{\mu}_{i,c}^\top\boldsymbol{\Sigma}_{i,c}^{-1}\boldsymbol{\mu}_{i,c})\right) dt \tag{24}$$

$$= \rho_i \cdot \int_0^{+\infty} \exp\left(-\frac{1}{2}(at^2 - 2bt + e)\right) dt, \tag{25}$$

$$\text{where} \qquad a = \boldsymbol{d}^\top\boldsymbol{\Sigma}_{i,c}^{-1}\boldsymbol{d}, \quad b = \boldsymbol{d}^\top\boldsymbol{\Sigma}_{i,c}^{-1}\boldsymbol{\mu}_{i,c}, \quad e = \boldsymbol{\mu}_{i,c}^\top\boldsymbol{\Sigma}_{i,c}^{-1}\boldsymbol{\mu}_{i,c}. \tag{26}$$

Substituting t with $u = \sqrt{a}(t - b/a)$ into the integral yields:

$$p_i(\boldsymbol{r}_c) = \rho_i \cdot \int_0^{+\infty} \exp\left(-\frac{a}{2}\left(t - \frac{b}{a}\right)^2 - \frac{e}{2} + \frac{b^2}{2a}\right) dt \tag{27}$$

$$= \rho_i \cdot \exp\left(-\frac{e}{2} + \frac{b^2}{2a}\right) \int_0^{+\infty} \exp\left(-\frac{a}{2}\left(t - \frac{b}{a}\right)^2\right) dt \tag{28}$$

$$= \rho_i \cdot \frac{1}{\sqrt{a}} \exp\left(-\frac{e}{2} + \frac{b^2}{2a}\right) \int_{-\frac{b}{\sqrt{a}}}^{+\infty} \exp\left(-\frac{u^2}{2}\right) du \tag{29}$$

$$= \rho_i \cdot \sqrt{\frac{\pi}{2a}} \exp\left(-\frac{e}{2} + \frac{b^2}{2a}\right) \left(1 + \mathrm{erf}\left(\frac{b}{\sqrt{2a}}\right)\right), \tag{30}$$

where erf(·) represents the error function. This expression is mathematically equivalent to Eq. (26) in [11]. In work [40] a simplified version at Eq. (20) is obtained, which assumes that the Gaussian is located at the origin of the coordinate system.

The computational complexity of the ray tracing method is much higher than our Exact-GS. For example, to compute the derivative of a:

$$\frac{\partial L}{\partial a} = \sum_{i=1}^{N} \frac{\partial L}{\partial p_i(\boldsymbol{r}_c)} \frac{\partial p_i(\boldsymbol{r}_c)}{\partial a}, \tag{31}$$

we must accumulate the gradients from the error function part $H(a,b)$, the exponential part $E(a,b,e)$ and the square root part $A(a)$:

$$A(a) = (2a)^{-1/2}, \quad E(a,b,e) = \exp\left(-\frac{e}{2} + \frac{b^2}{2a}\right), \quad H(a,b) = 1 + \text{erf}\left(\frac{b}{\sqrt{2a}}\right). \tag{32}$$

According to the chain rule of differentials, we can obtain:

$$\begin{aligned} &\frac{\partial p_i(\boldsymbol{r}_c)}{\partial a} = \rho_i\sqrt{\pi} \cdot \frac{\partial[A(a) \cdot E(a,b,e) \cdot H(a,b)]}{\partial a} \\ &= \rho_i\sqrt{\pi} \cdot \left[\frac{\partial A}{\partial a}E(a,b,e)H(a,b) + A(a)\frac{\partial E}{\partial a}H(a,b) + A(a)E(a,b,e)\frac{\partial H}{\partial a}\right], \\ &\text{where} \quad \frac{\partial A}{\partial a} = -(2a)^{-3/2}, \quad \frac{\partial E}{\partial a} = -\frac{b^2}{2a^2}E, \quad \frac{\partial H}{\partial a} = -\frac{b}{\sqrt{2\pi}}a^{-3/2}\exp\left(-\frac{b^2}{2a}\right). \end{aligned} \tag{33}$$

The expression for $\frac{\partial p_i(\boldsymbol{r}_c)}{\partial a}$ can be further expanded as:

$$\begin{aligned} &\frac{\partial p_i(\boldsymbol{r}_c)}{\partial a} = -(2a)^{-3/2}\rho_i\sqrt{\pi} \cdot \left[EH + \frac{b^2}{a}EH + b\sqrt{\frac{2}{\pi a}}\exp\left(-\frac{e}{2}\right)\right] \\ &= -(2a)^{-3/2}\rho_i\sqrt{\pi} \cdot \left[\left(1 + \frac{b^2}{a}\right)\exp\left(-\frac{e}{2} + \frac{b^2}{2a}\right)\left(1 + \text{erf}\left(\frac{b}{\sqrt{2a}}\right)\right) + b\sqrt{\frac{2}{\pi a}}\exp\left(-\frac{e}{2}\right)\right]. \end{aligned} \tag{34}$$

When computing the gradient of covariance matrix, it is necessary to collect all the gradient contributions of the three parameters $a(\boldsymbol{\Sigma}_{i,c}), b(\boldsymbol{\mu}_{i,c}, \boldsymbol{\Sigma}_{i,c})$, $e(\boldsymbol{\mu}_{i,c}, \boldsymbol{\Sigma}_{i,c})$:

$$\frac{\partial L}{\partial \boldsymbol{\Sigma}_{i,c}} = \frac{\partial L}{\partial a}\frac{\partial a}{\partial \boldsymbol{\Sigma}_{i,c}} + \frac{\partial L}{\partial b}\frac{\partial b}{\partial \boldsymbol{\Sigma}_{i,c}} + \frac{\partial L}{\partial e}\frac{\partial e}{\partial \boldsymbol{\Sigma}_{i,c}}. \tag{35}$$

The derivatives $\partial a/\partial \boldsymbol{\Sigma}_{i,c}, \partial b/\partial \boldsymbol{\Sigma}_{i,c}, \partial e/\partial \boldsymbol{\Sigma}_{i,c}$ requires the matrix multiplication to be accumulated three times, which has much higher computational complexity than our Exact-GS approach. RaySplats has a similar derivation of the gradient calculation to the ray tracing method [6].

8.5. Derivation of Corrected Splitting Scale Factor

The splitting algorithm from 3D-GS replaces a big Gaussian in the over-reconstruction region by two small ones. The scale ratio between the small one and original one is $5/8$, which is determined through experiments [17]. Other parameters are inherited from the original big Gaussian. To get a better analytical scale ratio, which can keep the projection unchanged after splitting, we calculate the total integral of Gaussian kernels with scale factor α. The integral of the big Gaussian kernel should be exactly twice that of the small kernel. We assume that the projection to the Gaussian is orthogonal, so the complete three-dimensional integral for the Gaussian kernel in Eq. (1) can be expressed as:

$$\begin{aligned} \mathcal{K}_i &= \int\int\int_V G_i(\boldsymbol{x}|\rho_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)dx_0dx_1dx_2 \\ &= \int_V \rho_i \cdot \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_i)\right)d\boldsymbol{x} \\ &= \rho_i(2\pi)^{3/2}|\boldsymbol{\Sigma}_i|^{1/2}. \end{aligned} \tag{36}$$

For the smaller Gaussian kernel, which has a scale factor α to the original one, the covariance matrix can be obtained:

$$\boldsymbol{\Sigma}_i^\alpha = \boldsymbol{R}_i(\alpha\boldsymbol{S}_i)(\alpha\boldsymbol{S}_i^T)\boldsymbol{R}_i^T = \alpha^2\boldsymbol{\Sigma}_i. \tag{37}$$

The corresponding relationship for determinant is:

$$|\boldsymbol{\Sigma}_i^\alpha| = |\alpha^2\boldsymbol{\Sigma}_i| = \alpha^6|\boldsymbol{\Sigma}_i|. \tag{38}$$

The integral of the big Gaussian kernel should be exactly twice that of the small kernel. Subsequently, we can compute the scale factor α:

$$\begin{aligned} \mathcal{K}_i &= 2\mathcal{K}_i^{\alpha}, \\ |\mathbf{\Sigma}_i|^{1/2} &= 2\alpha^3 |\mathbf{\Sigma}_i|^{1/2}, \\ \alpha &= 1/2^{1/3} \approx 0.7937. \end{aligned} \tag{39}$$

Although this has a small impact on the overall ADC and its improvement is difficult to verify experimentally, it was not mentioned in the main paper. However, it still contributes to a more accurate estimate of this parameter.

9. Implementation Details

9.1. Ray Tracing Renderer

For our reference projection generator, we developed a ray tracing renderer. Based on the mathematical derivation at Eq. (30), we wrote a kernel function that can run in parallel on the GPU, so we can render a large number of Gaussians and obtain accurate reference projection for synthetic dataset.

9.2. Exact-GS CUDA-based differentiable Rasterizer

We manually implemented forward and backward CUDA kernels in C++ for our proposed Exact-GS, where the backward kernel will produce all gradients of learnable parameters. Following the tile-based parallelization of 3D-GS, we use the same strategy: first separate all Gaussian kernels into corresponding tiles using an approximated projection. During the rendering each pixel is fixed in camera space, so the rotation matrix $\boldsymbol{T}$ remains unchanged. This can provide certain parallelization advantages. Our method and R2-GS [38] both define the minimum cutoff value as 10^{-8}, and projections smaller than this value will not be saved.

9.3. More on Initialization and Optimization

For synthetic Data. Since there is currently no effective solution to the problem of controlling the number of Gaussians, and each Gaussian's projection is constrained to a certain range, if this range does not intersect with the range of the true reference projection, gradients cannot be collected, and thus the parameters cannot be updated. Therefore, in the synthetic data, to ensure that the projection of the initialized Gaussian falls on the true reference projection range, the Gaussian is initialized to the reference parameter plus a small bias.

For real-world Data. When we evaluate the methods with real-world datasets, we use FDK [12] to obtain volume from sparse view projections. These voxels are then down-sampled according to a threshold to obtain 50k Gaussians for initialization, following R2-GS [38]. To adaptively control the number and density of Gaussians, we adopt the spawning and pruning heuristics from 3D-GS [17].

9.4. Loss Function

We optimized our model with a compound loss function:

$$\mathcal{L}_{all} = \mathcal{L}_2 + \lambda \mathcal{L}_{D-SSIM}, \tag{40}$$

where $\mathcal{L}_{D-SSIM}$ is the structural similarity loss between the rendered projection and the ground truth projection, $\mathcal{L}_2$ is the squared error loss, $\lambda = 0.25$. We adopt the same default optimization parameters as R2-GS [38]. .

9.5. Evaluation Metrics

We used metrics LPIPS [39], PSNR, and SSIM [33] to evaluate the reconstruction (3D) and projection (2D) quality.

9.6. Dataset Details

For each object, we choose 100 projections for metric evaluations. 75/50/25 projections are used to train the Gaussians. Due to positional accuracy issues for the misaligned center of rotation in the scanner, FIPS [29] recommended empirically translating the projection by a certain number of pixels. The difference image of 0-degree and 360-degree projections at Fig. 8 reveals relative motion of the object during scan, which negatively impacts reconstruction quality.

We resampled the projection using a rotation axis correction method. The resampled projection data significantly improved quality and reduced artifacts after reconstruction, see Fig. 9. Nevertheless, due to the large relative motion of the object during the scanning process, there is still a big difference between the 0° projection and the 360° projection. Subsequently, we similarly used the corrected 720 projections to perform FDK reconstruction to obtain the ground truth reference volume.

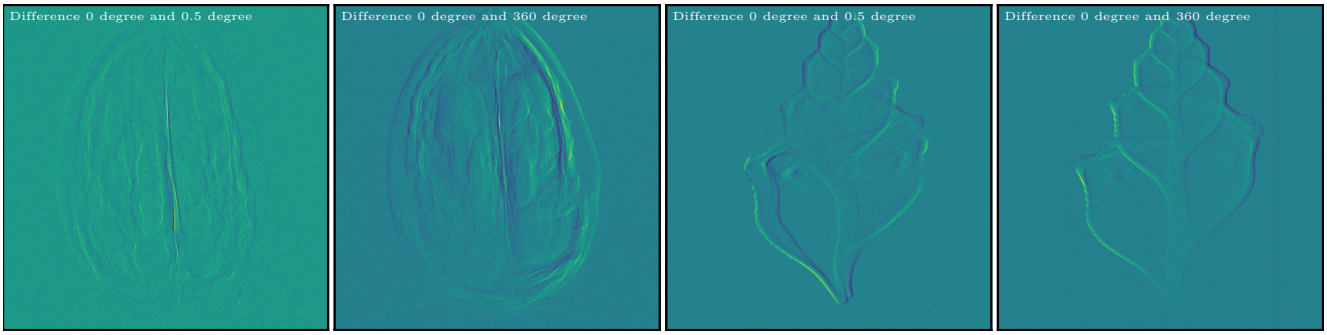


Figure 8. Visualization of the motion artifacts in the projection from FIPS [29]. The colormap represents increasing values from low (dark blue) to high (yellow). We show the difference of 0 and 360 degree projections in the second and fourth columns, which should theoretically be zero. It's clear that the projection error is large, indicating a noticeable shift between 0° and 360°. The first and third columns show the difference between the 0° and 0.5° projections of two objects, which is due to the small step rotation during the CT scan.

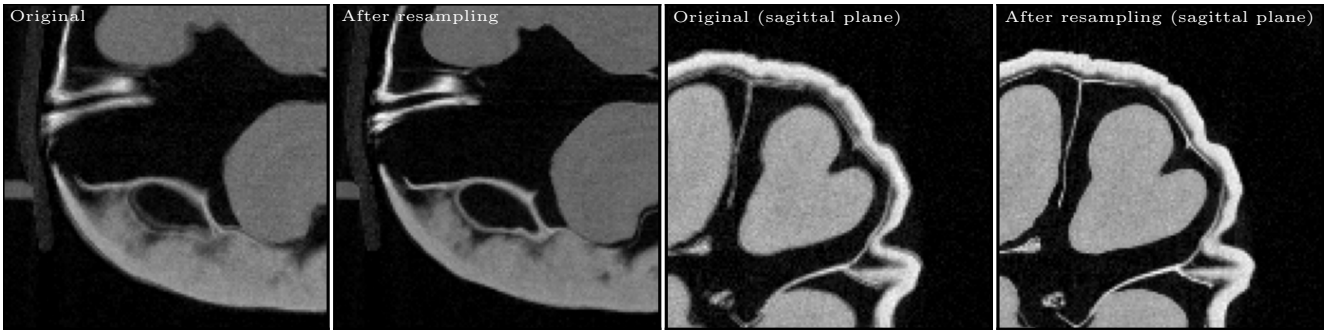


Figure 9. Comparison of FDK reconstructions using all 720 projections of FIPS [29] before and after center of rotation correction. All 720 projections are used for reconstruction. It can be clearly seen from the figure that after resampling, the artifacts caused by the wrong geometric position disappear and the overall image becomes clearer. The ghosting of thin strip structures is significantly reduced.

10. Additional Experiments

10.1. Additional Results for Convergence Study

In order to analyze the convergence properties of these two Gaussian Splatting methods on dataset with more Gaussians, we first cut off a small part with 912 Gaussians from a large Gaussian representation reconstruction. We initialize the Gaussians as a ball inside each ground truth Gaussian. Fig. 10 shows the results PSNR-2D and PSNR-3D during training.

We further evaluate the model's convergence properties across various numbers of views on a synthetic dataset with 5k Gaussians. Tab. 8 presents the quantitative results for novel view projection synthesis and volume reconstruction. In particular, when the number of projections decreases, our method can still maintain accurate reconstruction quality. From 75 to 25 projections, PSNR-3D only decreases by 5.2 dB. However, with R2-GS [38] it has decreased by 17.6 dB.

Table 8. Quantitative results of novel view synthesis and volume reconstruction using a synthetic sparse view dataset with 5000 Gaussians. We use LPIPS-2D* = LPIPS-2D $\times 10^6$.

75 Views						50 Views					
Method	PSNR-2D ↑	SSIM-2D ↑	LPIPS-2D* ↓	PSNR-3D↑	SSIM-3D ↑	Method	PSNR-2D ↑	SSIM-2D ↑	LPIPS-2D* ↓	PSNR-3D↑	SSIM-3D ↑
FDK [12]	30.78	0.865821	103838	26.78	0.546719	FDK [12]	27.76	0.816390	150505	23.72	0.461770
R2-GS [38]	64.28	0.999778	215.95	59.19	0.999828	R2-GS [38]	64.15	0.999780	215.24	57.86	0.999774
Ours	78.76	0.999998	7.54	65.50	0.999969	Ours	78.41	0.999998	7.14	63.61	0.999948
25 Views						10 Views					
Method	PSNR-2D ↑	SSIM-2D ↑	LPIPS-2D* ↓	PSNR-3D↑	SSIM-3D ↑	Method	PSNR-2D ↑	SSIM-2D ↑	LPIPS-2D* ↓	PSNR-3D↑	SSIM-3D ↑
FDK [12]	23.60	0.728273	235774	19.22	0.350278	FDK [12]	20.81	0.642682	319661	14.14	0.247797
R2-GS [38]	63.42	0.999767	264.85	53.00	0.999381	R2-GS [38]	56.40	0.999406	1483.20	41.55	0.993713
Ours	76.37	0.999998	8.70	64.16	0.999963	Ours	73.50	0.999995	30.40	60.30	0.999917

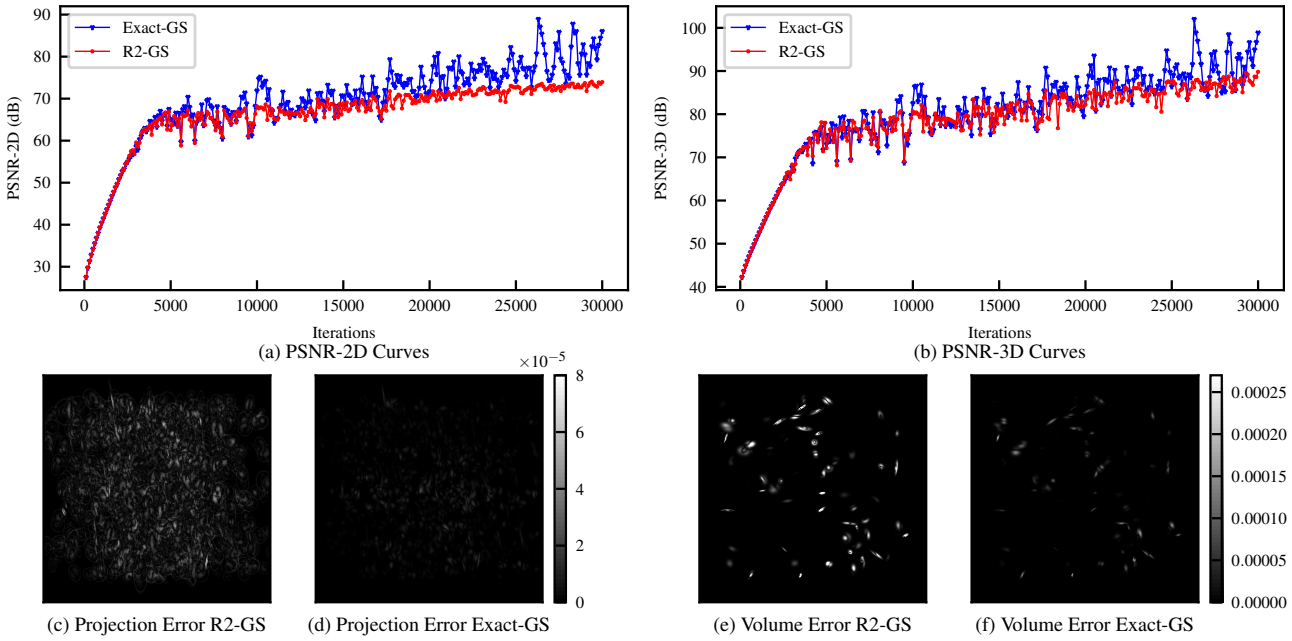


Figure 10. Comparison of reconstruction quantity and rendering quality between our Exact-GS and R2-GS [38]. (a) and (b) present the results for novel view projection synthesis (PSNR-2D) and volume reconstruction (PSNR-3D) as the number of iterations increases in th 912 Gaussians dataset. After 5000 iterations our method achieved much higher PSNR. We show slices of the absolute error maps for projection and volume after 30000 iterations in (c,d) and (d,f). We demonstrated that our method is able to consistently learn an object with less error than R2-GS [38].

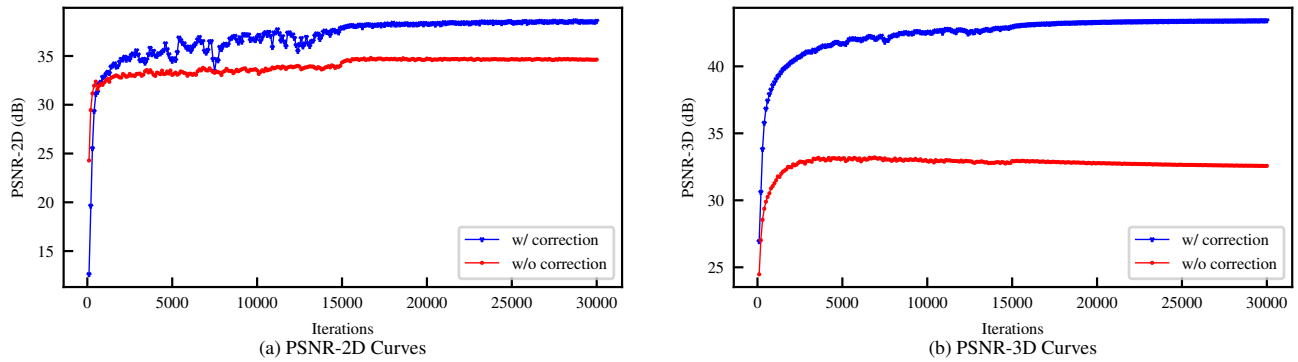
10.2. Additional Results for Real-World Dataset

First, we verified the reconstruction results of real-world data before and after center of rotation correction. As can be seen from Tab. 9, the FDK results after correction are significantly higher than before correction, and the PSNR-3D increased by nearly 10 dB. Before the correction, the PSNR difference between our method and R2-GS [38] was only 0.03 dB, and after the correction, it reached a 0.1 dB advantage. We present the PSNR curves during reconstruction with R2-GS [38] for real-world datasets before and after correction in Fig. 11. The data inconsistency from errors in the geometric parameter causes the PSNR value in the reconstruction process to first increase and then even decrease. After correcting the data, this phenomenon disappeared.

Table 9. Quantitative results for comparison of center of rotation correction on the real-world dataset.

	75 Views				50 Views				25 Views			
Method and Data	PSNR-2D	SSIM-2D	PSNR-3D	SSIM-3D	PSNR-2D	SSIM-2D	PSNR-3D	SSIM-3D	PSNR-2D	SSIM-2D	PSNR-3D	SSIM-3D
FDK [12] w/o Correction	22.78	0.497	25.12	0.380	21.02	0.455	22.76	0.298	18.27	0.385	18.94	0.199
R2-GS [38] w/o Correction	34.60	0.927	32.56	0.710	34.14	0.918	30.58	0.687	30.79	0.884	26.63	0.637
Ours w/o Correction	34.63	0.927	32.52	0.701	34.12	0.917	30.50	0.677	30.55	0.883	26.52	0.625
FDK [12] w/ Correction	23.35	0.524	35.62	0.723	21.46	0.481	33.31	0.630	18.54	0.406	29.32	0.479
R2-GS [38] w/ Correction	38.99	0.954	44.02	0.946	38.31	0.949	42.69	0.940	33.89	0.915	38.10	0.909
Ours w/ Correction	39.11	0.955	44.11	0.944	38.40	0.949	42.69	0.938	33.99	0.915	38.11	0.906

Then we further evaluate all methods on the corrected real-world datasets in Fig. 12. In the 75-projection dataset, although the PSNR-3D of our method is second only to NAF [37], it can be seen from the error maps that our method can reconstruct the high-frequency structure in the volume more clearly. As the number of projections decreases, it can be observed that volumes from both 3D-GS methods have striped artifacts in the smooth low-frequency structure area, which reduces the PSNR-3D significantly. However, for some high-frequency structures of the outer contour, the reconstruction results of the 3D-GS methods are still better than NeFR-based methods.



(a) PSNR-2D Curves (b) PSNR-3D Curves

Figure 11. Comparison of center of rotation correction from R2-GS [38] reconstruction. (a) and (b) present the results for novel view projection synthesis (PSNR-2D) and volume reconstruction (PSNR-3D) in real-world datasets. It can be clearly seen that in the uncorrected data, PSNR-3D shows an abnormal decrease with more training iterations. In the corrected data, both PSNR-2D and PSNR-3D increase slowly and monotonically.

10.3. Additional Results for Synthetic Dataset

This experiment evaluates our method on more synthetic datasets. In Fig. 13 and Fig. 14, we visualize the error map of the reconstructed volume from our Exact-GS and R2-GS [38]. In the case of very sparse projections, due to limited boundary conditions, both methods cannot converge very well, so the gap between them becomes smaller. It is clear that our method achieved higher accuracy across all different views.

In addition, we tested our method using synthetic data based on voxel grids ground truth volume. The projection are generated with tomography toolbox TIGRE [3]. We visualize the reconstructed volume from our Exact-GS and NAF [37] at Fig. 15. For porous structures in cement, there are numerous high-frequency edge structures within the volume. Our Exact-GS achieves a 1.64 dB improvement over the NAF [37]. This further validates the advantage of our method for high-frequency structures.

11. Additional Discussion

11.1. Impact of Numerical Precision

A single-precision float (32 bits) has a precision of approximately 7 decimal places. R2-GS [38] truncated all projection values less than 10^{-5}. To verify the accuracy of our method, we conducted experiments with different truncation values on a single Gaussian synthetic dataset. As can be seen from the results in Tab. 10, R2-GS [38] produces essentially the same results at two different precision levels. However, our method, using CTAccuTile algorithm, significantly improves projection quality, proving the validity of our approach.

Table 10. Numerical Precision study results. We evaluate the model in terms of projection quality (2D) and volume reconstruction quality (3D) for different truncation thresholds.

Method	Threshold	MSE-2D ↓	PSNR-2D ↑	MSE-3D ↓	PSNR-3D↑
R2-GS [38]	10^{-8}	3.54×10^{-7}	66.36	1.48×10^{-9}	88.27
Exact-GS (Ours)	10^{-8}	2.50×10^{-15}	147.28	2.96×10^{-16}	155.26
R2-GS [38]	10^{-5}	3.54×10^{-7}	66.36	1.50×10^{-9}	88.23
Exact-GS (Ours)	10^{-5}	3.58×10^{-11}	104.52	2.98×10^{-16}	155.25

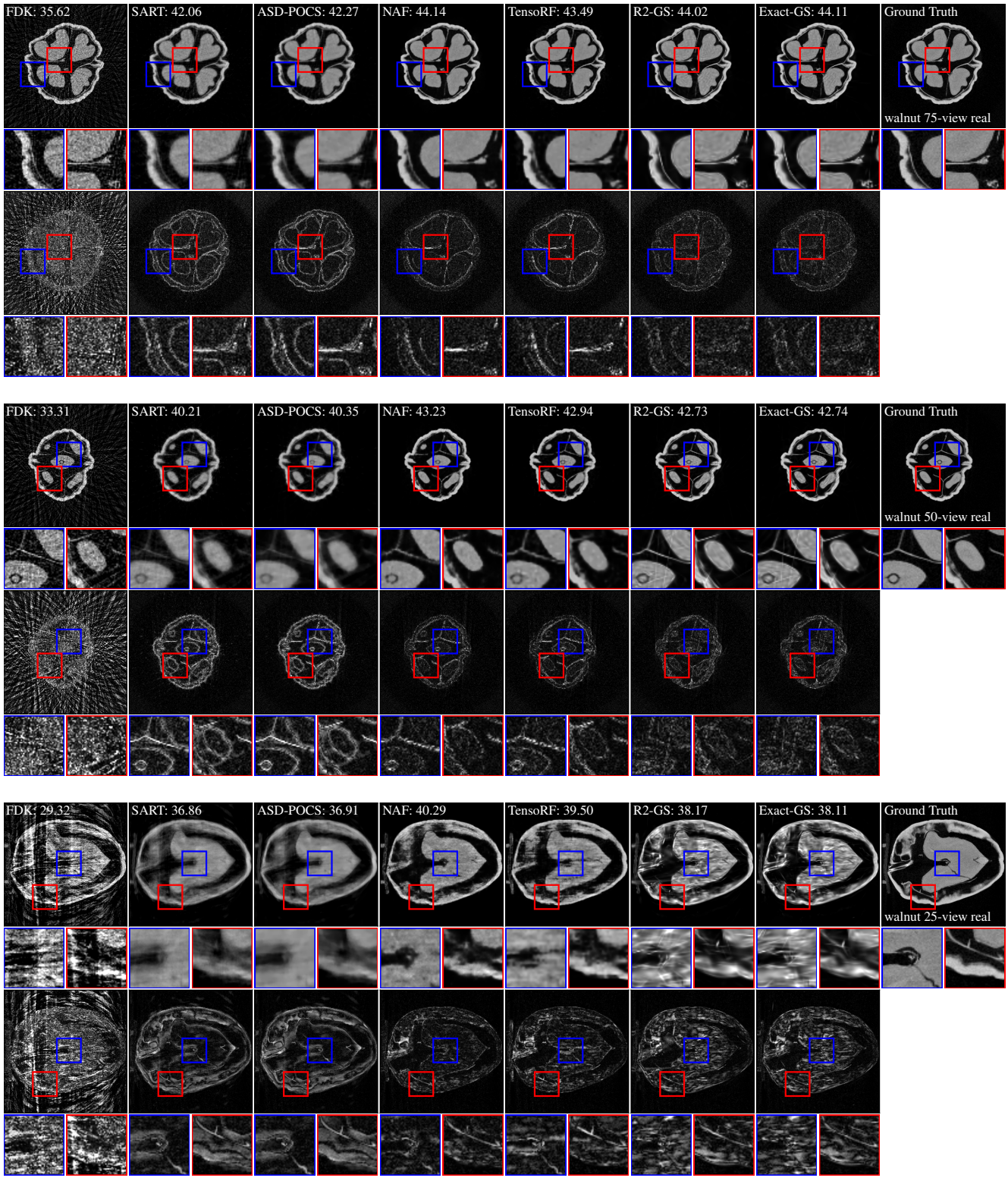


Figure 12. Additional qualitative comparisons on corrected real-world dataset for volume reconstruction. PSNR-3D (dB) are shown at the top of each volume slice. Absolute error maps are illustrated under each image, with darker colors representing smaller values.

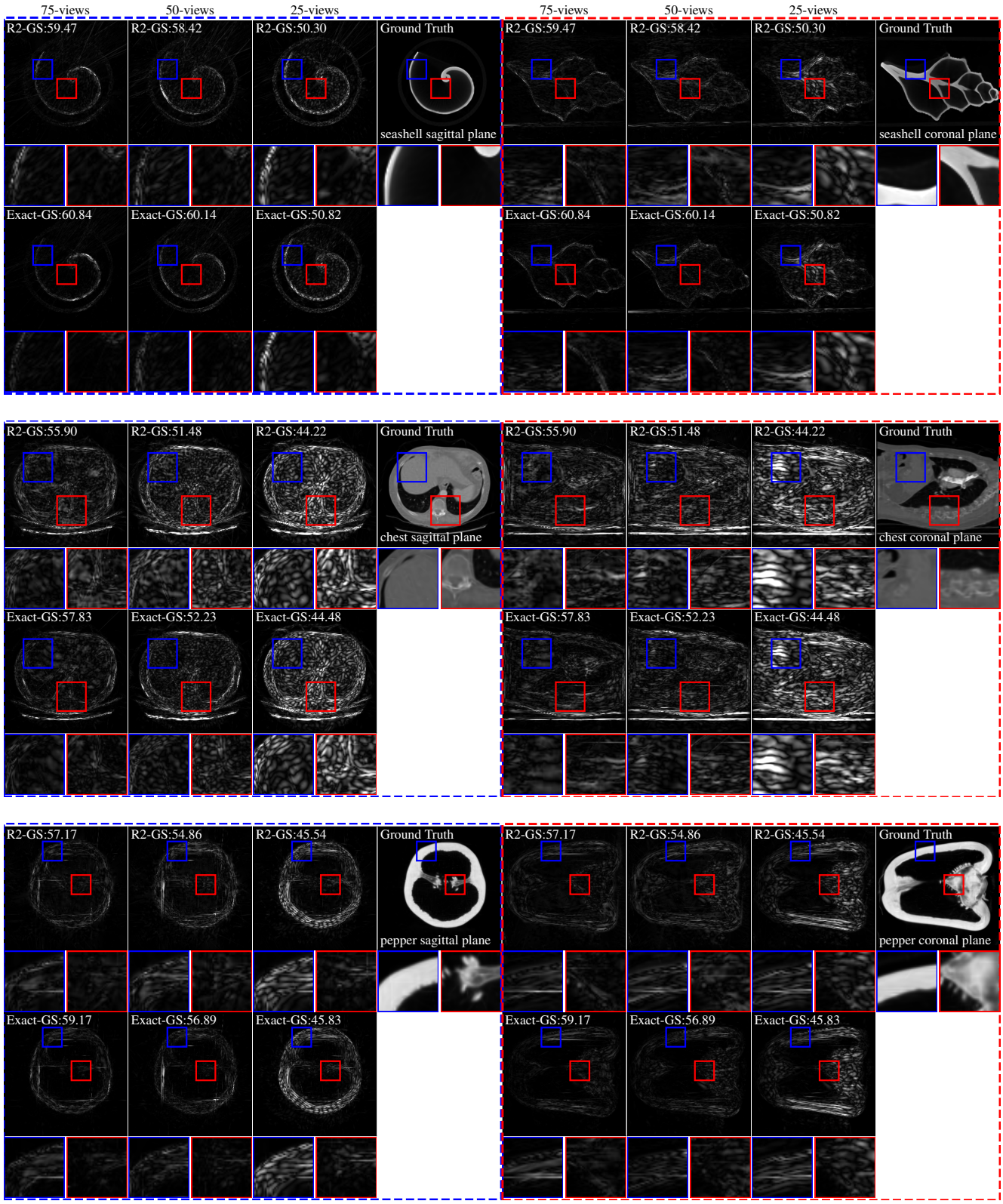


Figure 13. Qualitative comparison between our Exact-GS and R2-GS [38] on synthetic dataset for volume reconstruction. PSNR-3D (dB) are shown at the top of each absolute error map. Part I.

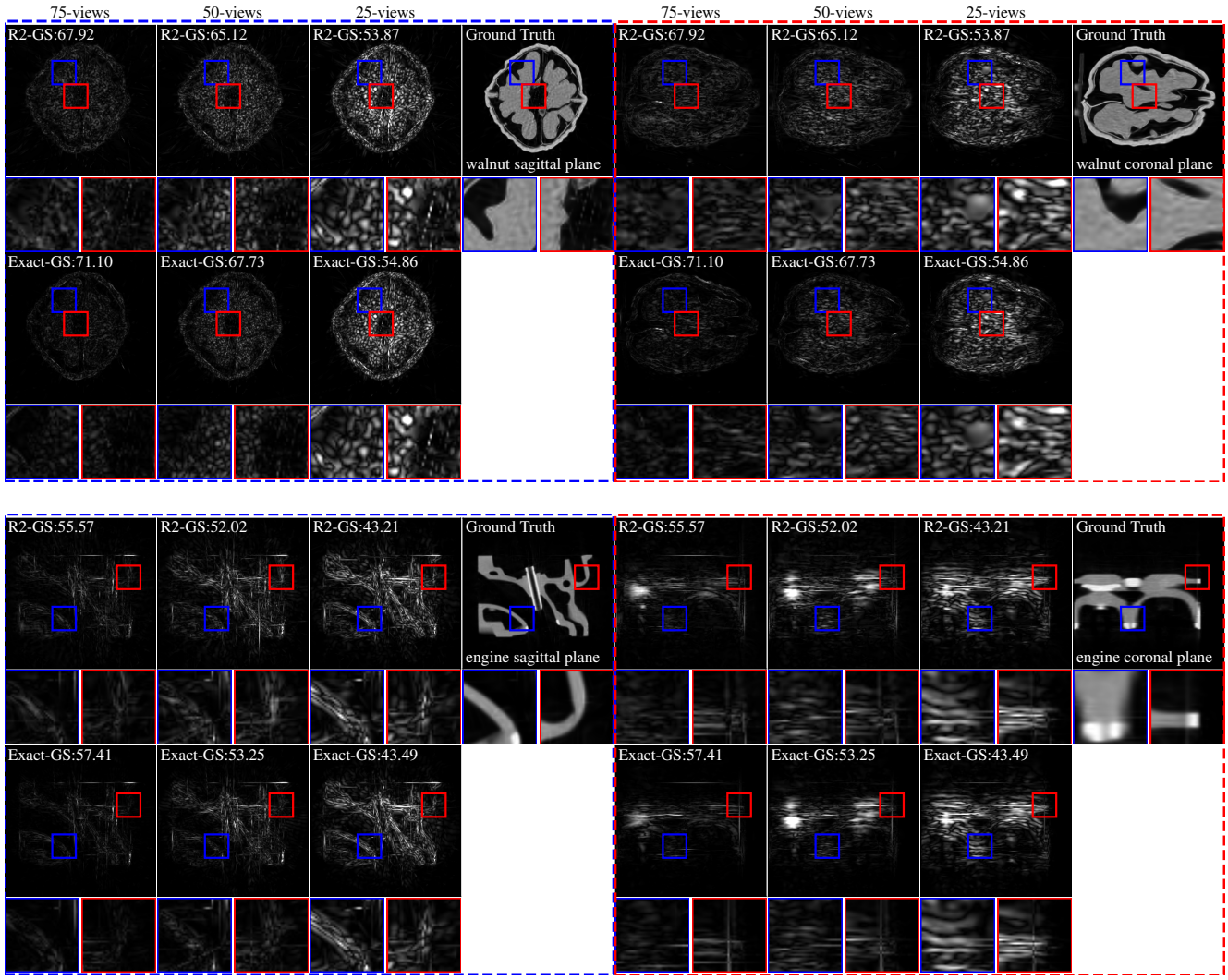


Figure 14. Qualitative comparison between our Exact-GS and R2-GS [38] on synthetic dataset for volume reconstruction. Part II.

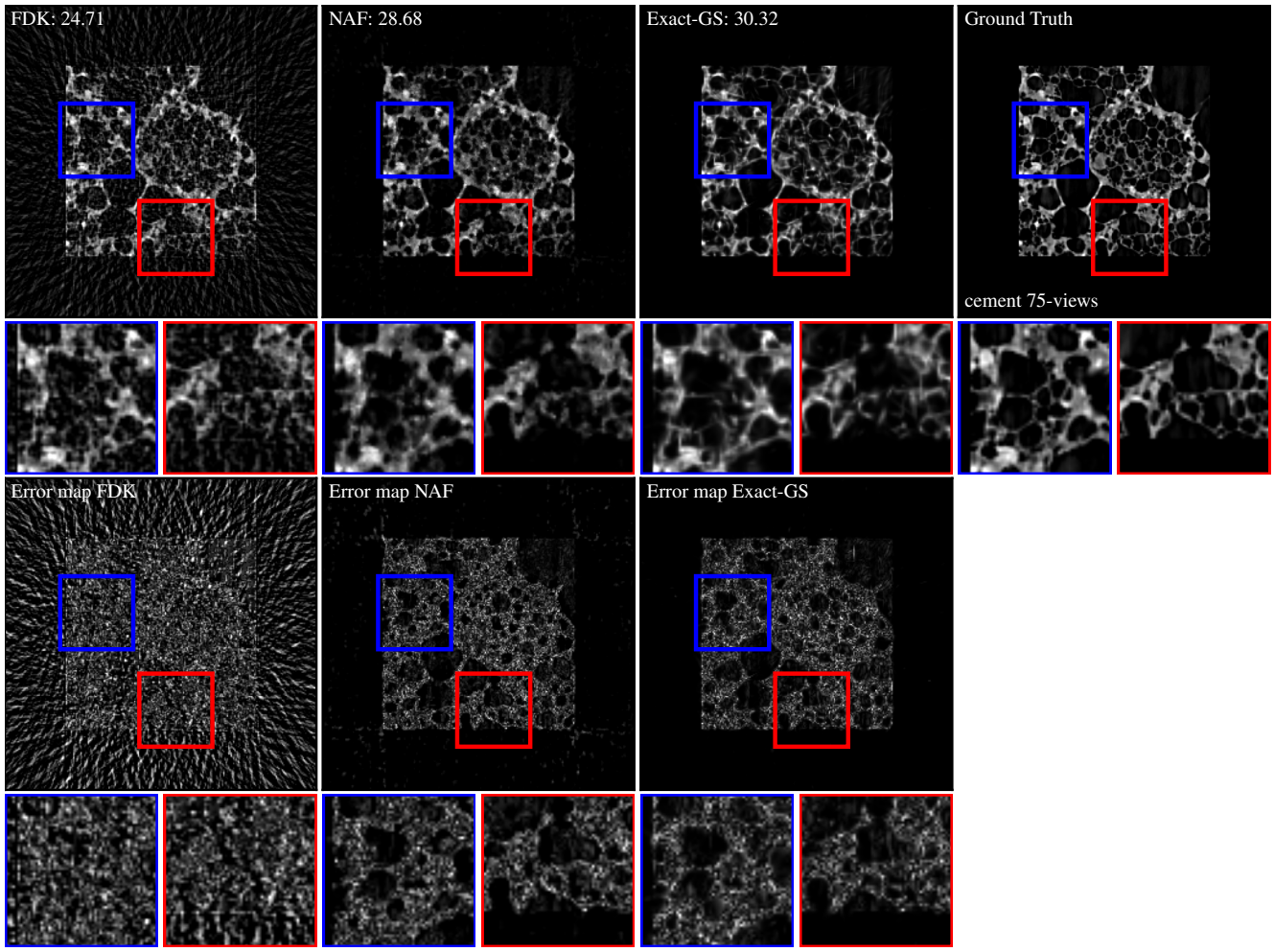


Figure 15. Qualitative comparison between our Exact-GS and NAF [37] on voxel-based synthetic dataset for volume reconstruction. PSNR-3D (dB) are shown at the top of each image.