Z-Splat: Z-Axis Gaussian Splatting for Camera-Sonar Fusion

Ziyuan Qu*, Omkar Vengurlekar†, Mohamad Qadri‡, Kevin Zhang§, Michael Kaess‡, Christopher Metzler§, Suren Jayasuriya†, and Adithya Pediredla*
*Dartmouth College, †Arizona State University, ‡Carnegie Mellon University, §University of Maryland

Abstract—Differentiable 3D-Gaussian splatting (GS) is emerging as a prominent technique in computer vision and graphics for reconstructing 3D scenes. GS represents a scene as a set of 3D Gaussians with varying opacities and employs a computationally efficient splatting operation along with analytical derivatives to compute the 3D Gaussian parameters given scene images captured from various viewpoints. Unfortunately, capturing surround view (360° viewpoint) images is impossible or impractical in many real-world imaging scenarios, including underwater imaging, rooms inside a building, and autonomous navigation. In these restricted baseline imaging scenarios, the GS algorithm suffers from a well-known 'missing cone' problem, which results in poor reconstruction along the depth axis. In this paper, we demonstrate that using transient data (from sonars) allows us to address the missing cone problem by sampling high-frequency data along the depth axis. We extend the Gaussian splatting algorithms for two commonly used sonars and propose fusion algorithms that simultaneously utilize RGB camera data and sonar data. Through simulations, emulations, and hardware experiments across various imaging scenarios, we show that the proposed fusion algorithms lead to significantly better novel view synthesis (5 dB improvement in PSNR) and 3D geometry reconstruction (60% lower Chamfer distance).

Index Terms—Gaussian Splatting, Acoustic-Optic Vision, Sensor fusion.

1 Introduction

Differentiable Gaussian spatting (GS) [1], a resurgence of elliptical weighted average (EWA) splatting [2] from almost two decades back, has been emerging as the dominant differentiable representation for scenes [3, 4]. GS represents the scene as a volume density map similar to neural radiance fields (Nerf) [1, 5]. However, unlike Nerfs, GS explicitly represents the scene as a sum of densities of anisotropic 3D Gaussians. The explicit representation facilitates easier geometric interpretation and manipulation of the scene, leading to an explosion of GS-based techniques for scene relighting [6], shading [7], novel view synthesis [8], text-based manipulation [9], and non-rigid manipulation [10]. Furthermore, the GS rendering algorithm is fast, thanks to the splatting operator and analytical derivatives which leads to extensions to dynamic scenes [11, 12], 4D manipulation [13–15] and time-efficient editing [9, 16].

Unfortunately, when the training dataset has a limited baseline, optimizing through the GS algorithm causes overfitting leading to poor novel-view rendering and 3D reconstruction. In Section 3.4, we show that limited baseline results in the well-known 'missing cone' problem [17, 18] where the frequencies along the depth-axis are not captured in the training data. To mitigate the problem of missing depth information from the training images, Chung et al. [19] proposed regularizing 3D Gaussian splatting with monocular depth estimates. However, as this method hallucinates depth, the scene reconstructions are no longer physically based and suffer from training bias. In this paper, we use sonar images to sample the missing cone region.

Recent progress in sonar technologies has resulted in several low-cost sensors that augment 2D spatial data measured by RGB cameras. Examples include echosounder [20], forward-looking sonar [21], synthetic aperture sonar [22] which find applications in SLAM [23], navigation [24, 25], underwater imaging [26], and imaging through scattering media [27]. These sonars offer complementary information compared to the standard RGB cameras.

In this paper, we extend Gaussian splatting for sonar and build fusion techniques that reconstruct geometry using the complementary information from both the cameras and sonars. Our extension involves the development of splatting operations along the z-axis tailored for these sensor types. Through simulations, emulations, and hardware experiments, we show that combining sonars and cameras results in significantly better (60%) 3D geometry reconstruction. We also show that the accuracy of novel view synthesis improves by 5 dB by regularizing camera data with sonars.

The specific contributions of this paper include

1) A novel forward model to render the transient of Gaussian point clouds for two types of sonars: Echosounder and Forward-Looking Sonar (FLS).
2) Fusion algorithms for cameras and sonars.
3) Validation on synthetic, emulated hardware, and real hardware datasets showing that fusion GS splatting results in better geometric (60%) and photometric (5 dB) reconstruction than standard camera-only Gaussian splatting.

We release our source code and data to the larger community [28]. We hope that our work inspires the use of complementary sensor modalities to improve 3D scene representation in the future.

2 Related Work

Gaussian splatting: In computer graphics and rendering, splatting algorithms were introduced over two decades

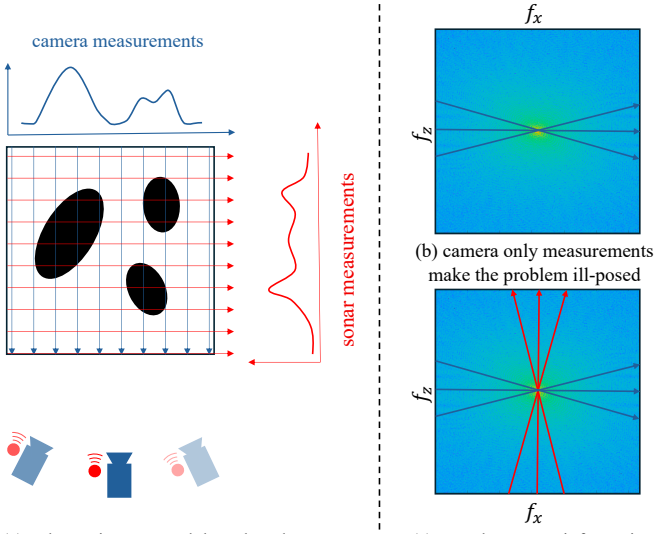
ago [2] for texture filtering [29] and point cloud rendering [30, 31], where the scene is represented as a sum of anisotropic Gaussian kernel that can be efficiently rendered and without aliasing artifacts.

The Gaussian splatting paper [1] uses this scene representation and fast differentiable rendering pipeline to compute the scene parameters (density, color, mean, and variance of the 3D Gaussians). Thanks to the decreased computation on empty spaces, analytical derivatives, rich-quality reconstructions, and explicit representations, Gaussian splatting has seen an explosion of extensions, even though it was introduced less than a year ago (see Fei et al. [32] for a review of the state-of-the-art). Related to this paper, Matsuki et al. [33], Keetha et al. [34], Yan et al. [35], Sun et al. [36] have independently proposed similar SLAM algorithms using RGB-D cameras. The key idea is to splat depth similar to color values and minimize the weighted sum of photometric and geometric loss functions. However, this idea cannot be extended to sonar data as these techniques require a single depth value per pixel, necessitating the dataset to be a depth map with high spatial resolution on the x-y plane. Instead, sonar data typically resembles a transient histogram of time-of-flight returns.

Camera and sonar fusion techniques: There has been previous research on fusing complementary information from sonars and cameras in the literature [37–39]. For example, optical cameras suffer from hazing and scattering problems, especially in turbid waters [37]. Sonars do not suffer from scattering, though they have poor spatial resolution. To address this, Raaj et al. [38] combined FLS and optical cameras for 3D object localization via particle filter in scattering environments. Williams and Mahon [39] combined an echosounder (also known as depth sounder) and a vision camera to develop a SLAM algorithm for underwater robotic navigation on the Great Barrier Reef. This paper focuses on the complementary geometric information that sonars and cameras provide and analyzes how the missing cone present in camera-based 3D reconstruction can be resolved via our Gaussian splatting algorithms for transient histograms to reconstruct better geometry and photometry.

Most related to our work, Babaee and Negahdaripour [40] reconstructed 3D objects from RGB and imaging sonars. However, their technique requires matching occluding contours across RGB and imaging sonar, which restricts its applicability to small baseline scenarios. Qadri et al. [21] recently propose combining FLS sonar and cameras using implicit neural representations (INR). Unlike ours, Qadri et al.'s technique cannot easily extend to echosounder and reconstructs only scene geometry, not photometry.

Other fusion techniques: In addition to sonar, researchers have proposed fusing radar measurements with radar for better object detection [41], imaging of cluttered environment [42], and estimating pose and motion of vehicles for autonomous navigation. Fusing Lidars or other optical time-of-flight cameras is similar to our problem and has been targeted at imaging through scattering media [43], densification of lidar point scans [44], and improved depth estimation [45–47]. The proposed Gaussian splatting algorithm could be extended to fusion with Radars and Lidars.



(a) volumetric scene and the splatted camera and sonar measurements

(b) camera only measurements make the problem ill-posed

(c) complementary information helps few-shot problem

Fig. 1. **Sonar measurements provide complementary information.** (a) Volumetric scene captured with three pairs of cameras and sonars (echosounder). We assume the sensors are in the far field (i.e., the affine approximation to the projective transform in Gaussian splatting research is valid). For the center camera-sonar pair, camera measurements are obtained by projecting the volumetric data along the vertical axis, and sonar measurements are obtained by projecting the volumetric data along the horizontal axis. (b) If only camera measurements are considered, then using the Fourier-slice theorem, we are capturing only a few slices of the Fourier transform of the volume and missing information on a large cone. (c) Sonar (time-resolved data) captures orthogonal slices in the Fourier space, and hence, 3D reconstruction of the scene is better conditioned if we do the camera-sensor fusion instead of using only camera data.

3 Technical Background

In this section, we will briefly review the scene representation, rendering equation, and splatting operation associated with the Gaussian splatting algorithm. We will show that traditional GS algorithms do not appropriately capture the z-related scene parameters, resulting in the missing cone problem. This problem motivates our camera and sonar fusion solution to recover this lost information.

3.1 Scene representation

In Gaussian splatting, the scene is represented as a volumetric density map. This density ($\sigma \in \mathbb{R}^+$) at a point ($\bar{x} \in \mathbb{R}^3$) is given as the sum of densities contributed by several Gaussians located as positions ($\bar{\mu}_n \in \mathbb{R}^3$) and 3×3 anisotropic variance (Σ_n). Analytically,

$$\sigma(\bar{x}) = \sum_{n=1}^{N} \sigma_n \mathcal{N}(\bar{x}; \bar{\mu}_n, \Sigma_n), \tag{1}$$

where σ_n is a scaling factor representing the density of each Gaussian. The color at each 3D point is similarly defined as

$$\bar{c}(\bar{x}) = \sum_{n=1}^{N} \bar{c}_n \mathcal{N}(\bar{x}; \bar{\mu}_n, \Sigma_n), \tag{2}$$

where $\bar{c}_n$ is the color of each Gaussian kernel, which is encoded using spherical harmonics [1].

The covariance of a 3D Gaussian is represented using the scaling matrix S_n and a rotation matrix R_n to maintain the positive definiteness as:

$$\Sigma_n = R_n S_n S_n^T R_n^T. \tag{3}$$

The rotation matrix R_n is calculated from a normalized quaternion q_n. For brevity, we will drop the index n wherever the index is implied.

3.2 Volume rendering equation

Gaussian splatting uses the same volume rendering equation as point-based α-blending [48] or NeRF-style [5] volumetric rendering. Therefore, if a ray ($\bar{x}(l) = \bar{o} + l\bar{d}$) is cast along a pixel on the camera, the color C of the pixel will be

$$C = \sum_{m=1}^{M} T_m \alpha_m c_m, \tag{4}$$

where the index m represents the m^{th} segment of the ray with small length δl, and

$$\alpha_m = 1 - e^{-\sigma(\bar{x})\delta l}, \;\; T_m = \prod_{k=1}^{m-1} (1 - \alpha_k), \text{and}$$
$$c_m = c(\bar{x}(m\delta_l)). \tag{5}$$

3.3 Splatting operation

Instead of expensive ray tracing similar to NeRF-style algorithms, the GS algorithms [1, 2] use splatting, which is similar to rasterization. For a camera viewing transform W, the 3D volume is transformed appropriately to the camera view, but the GS algorithm approximates the projection operation with an affine transformation so that any 3D Gaussian remains a 3D Gaussian with covariance matrix:

$$\Sigma' = JW\Sigma W^T J^T, \text{ and mean } \mu' = JW\mu \tag{6}$$

where J is the Jacobian for the local affine approximation for each Gaussian kernel

$$J = \begin{bmatrix} \frac{1}{\mu_z} & 0 & \frac{-\mu_x}{\mu_z^2} \\ 0 & \frac{1}{\mu_z} & \frac{-\mu_y}{\mu_z^2} \\ \frac{\mu_x}{l} & \frac{\mu_y}{l} & \frac{\mu_z}{l}, \end{bmatrix} \tag{7}$$

and where $\mu = [\mu_x, \mu_y, \mu_z]$ and $l = \|\mu\|_2$. The local affine approximation preserves the Euclidean distance from the camera to the object. Based on the equation (15) in the EWA Splatting [2], the transformed domain is

$$\begin{bmatrix} \mu'_x & \mu'_y & \mu'_z \end{bmatrix}^T = \begin{bmatrix} \mu_x/\mu_z & \mu_y/\mu_z & ||(\mu_x, \mu_y, \mu_z)^T|| \end{bmatrix}^T.$$

Note that the distances between the camera and Gaussian centers are still $||(\mu_x, \mu_y, \mu_z)^T||$. The visual illustration is shown in Figure 2 (a) and (b). This ensures the relative accuracy of the sonar physical model.

The next step after the approximate projection operation is orthographic projection. The 3D Gaussians are converted to 2D Gaussians with covariance matrix Σ'_{2D} obtained by dropping the last row and column, mathematically,

$$\Sigma'_{\text{2D}} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \Sigma' \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{8}$$

The 2D Gaussians are α-blended using Equation (4) to synthesize the camera image.

3.4 The Missing Cone

In small baseline imaging scenarios, information about the covariances (σ_{xz}, σ_{yz}) and variance (σ_{zz}) associated with the depth-axis and the mean (μ_z) along the depth-axis are not captured by the camera images.

To illustrate this, consider a simple volumetric imaging scenario in Figure 1. We consider 2D volume (the arguments are extendable to 3D volume) and assume no interreflections and far-field sensing modalities, i.e., volumetric scene after the local affine approximation in the GS algorithm. For these volumetric scenes, the rendered RGB image will be the x-axis projection (splat) of the volume.

Using the Fourier-slice theorem, the Fourier transform of the camera image is the same as the x-slice of the volume's Fourier transform. By taking several images from various angles, we are capturing various slices of the volume's Fourier transform. Unfortunately, for the small baseline scenarios [21], we have the missing cone problem [17, 18, 49], which refers to a cone in the Fourier space that was not sensed by the camera images. This limits the fidelity of the 3D reconstructions due to missing frequency information.

Our approach to solving this problem is to utilize complementary information present in time-resolved measurements, particularly time-resolved measurements from sonar, which can be obtained by taking the z-axis projection of the volume as we will discuss next in (Section 4). From the Fourier-slice theorem, this is equivalent to obtaining a z-slice of the volume's Fourier transform as shown in Figure 1(c). Hence, sonars can help capture the missing cone and augment camera data, particularly when camera baselines are limited.

4 Z-Axis Gaussian Splatting for Camera-Sonar Fusion

We will first extend the Gaussian splatting model to two commonly used sonars, i.e. echosounder and FLS, by exploiting the physics of how the sonars operate. After that, we will combine the camera and sonar Gaussian splatting to build a fusion algorithm capable of reconstructing the 3D Gaussian parameters accurately even for small baseline imaging scenarios.

4.1 Echosounder

A single-beam echosounder utilizes a single transducer element to emit a sound pulse toward the target area and captures the echo, a time-varying sound pulse that bounces back from the scene. This echoed signal is a function of acoustic reflectivity and depth of various scene points. For simplicity, we assume the acoustic albedo is constant for all the objects in the scene. Therefore, the echoed signal can also be interpreted as a 1D histogram of scene depths computed through acoustic time-of-flight.

In Figure 2, we illustrate the proposed splatting operation for echosounder. We splat the 3D Gaussians along the z-axis, while computing the transmission and alpha values similar to the camera splatting. The covariance of the 1D projection of 3D Gaussians is:

$$\Sigma'_{\text{1D}} = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \Sigma' \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \sigma_{zz}. \tag{9}$$

Algorithm 1 Echosounder rendering

$Z[d] = 0$;
for each point on the xy-plane **do**
  for each visible Gaussian Kernel **do**
    for each bin i within $\mu_z \pm 3\sigma_{zz}$ **do**
      $z =$ center of the bin
      $Z[i] += T \cdot \alpha \cdot \exp(-\frac{(z-\mu_z)^2}{2\sigma_{zz}})$
    end for
  end for
end for

Algorithm 2 FLS rendering

$Z[h, d] = 0$;
for each point on the xy-plane **do**
  for each visible Gaussian Kernel **do**
    for each bin i within $\mu \pm 3 \cdot \sigma_{zz}$ **do**
      $z =$ center of the bin
      $j =$ y-center of the Gaussian
      $Z[j, i] += T \cdot \alpha \cdot \exp(-\frac{(z-\mu_z)^2}{2\sigma_{zz}})$
    end for
  end for
end for

Therefore, the echosounder captures the information about mean μ_z and variance σ_{zz} along the depth axis that was missing in the camera data. The proposed splatting operation computes the depth histogram of the volumetric scene while accounting for the visibility term accurately.

We show the steps of the z-axis splatting for echosounder in Algorithm 1. We splat Gaussians for each xy-pixel during rasterization. Therefore, if the scene is represented as one large Gaussian or several small Gaussians, the rasterized Z-splat result would be same.

4.2 Forward Looking Sonar (FLS):

FLS is similar to an echosounder, except it contains multiple linear transducer elements. Through beamforming, FLS recovers the range and azimuth information but not the elevation. Therefore, in the orthographic view (Figure 2(b)), FLS measures depth histograms for various y values.

To build a rendering algorithm for FLS, we splat the 3D Gaussians on the $y-z$ plane, resulting in 2D Gaussians with covariance matrix:

$$\Sigma'_{2D} = \begin{bmatrix} \sigma_{yy} & \sigma_{yz} \\ \sigma_{yz} & \sigma_{zz} \end{bmatrix}. \tag{10}$$

We compute the transmission and alpha values similar to camera splatting, as the visibility terms do not change for FLS. We identify the steps in FLS rendering in Algorithm 2.

In practice, an RGB camera often has a higher spatial resolution than a sonar, so the information on the y-axis mainly comes from the RGB camera. Therefore, we expect the reconstruction accuracy and quality of novel synthetic views supervised by FLS to be slightly better than those from echosounder, but not significantly better.

4.3 Fusion of cameras and sonars

We fuse the camera and sonar information by jointly minimizing the error between the rendered and measured data

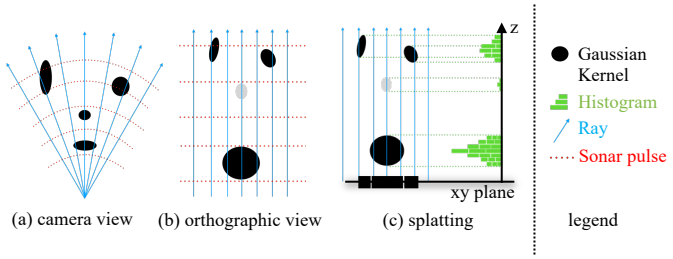


Fig. 2. **Ray View Transformation and Z-Axis Splatting** (a) This illustration shows the camera view. The covariance of Gaussians in the camera view is $\Sigma = W^T \Sigma W$, which transforms the Gaussians from the world view to the camera view. (b) The Gaussians are transformed into the ray view through an local affine approximation of the projection transform using the Jacobian (J). The covariance matrix of the Gaussians will be $\Sigma' = J^T \Sigma J$. (c) The transformed 3D Gaussian is then projected (splat) onto the xy-plane for rendering camera and z-axis for rendering echosounder (for collocated camera and echosounder). The gray Gaussian is occluded by the Gaussian in the front, so the Transmission(T) of that Gaussian is smaller than the others independent of whether we are rendering camera or sonar. Based on Algorithm 1 and Algorithm 2, each ray undergoes splatting independently, ensuring that if a Gaussian is rasterized by multiple rays, it will be splatted multiple times.

for both sensors. We define the camera loss $\mathcal{L}_c$ as:

$$\mathcal{L}_c = \|I(x, y) - I_{\text{gs}}(x, y)\|_1, \tag{11}$$

where $I(x, y)$ represents the measured camera image and $I_{gs}(x, y)$ is the rendered image from Gaussian splatting.

The sonar loss ($\mathcal{L}_s$) is defined as

$$\mathcal{L}_s = \begin{cases} \|S(z) - S_{\text{gs}}(z)\|_2, & \text{for Echosounder} \\ \|S(y, z) - S_{\text{gs}}(y, z)\|_2, & \text{for FLS.} \end{cases} \tag{12}$$

Here $S(z)$ and $S(y, z)$ are the measured echosounder and FLS data; S_{gs} and $S_{\text{gs}}(y, z)$ are the Gaussian splatted renderings discussed in Section 4.1 and Section 4.2. Note that L_s is a vector norm for echosounder and a Frobenius norm for FLS.

We define the loss function as a weighted combination of camera and sonar loss functions:

$$\mathcal{L} = \mathcal{L}_c + w \cdot \mathcal{L}_s. \tag{13}$$

Empirically, we found that setting $0.1 \leq w \leq 3$ is suitable for most scenarios. We can tune w based on the confidence between camera and sonar measurements. For instance, in Figure 4, we choose $w = 3$ for the living room scene because the scene does not have a lot of texture. We can also choose adaptive weighing schemes similar to Qadri et al. [21] where more importance is given to sonar measurements in the beginning and camera measurements in the later iterations, though we did not find that weighing scheme useful in this case.

4.4 Implementation

We have implemented the proposed algorithms by extending the publicly available Gaussian splatting source code shared by Kerbl et al. [1]. We computed the derivatives analytically and wrote cuda kernels to improve the rendering and training speed. The average training time for all the simulated experiments in the paper is around 5 minutes on a NVIDIA 4090 GPU, while real experiments took 20 minutes for echosounder and 8 minutes for FLS.

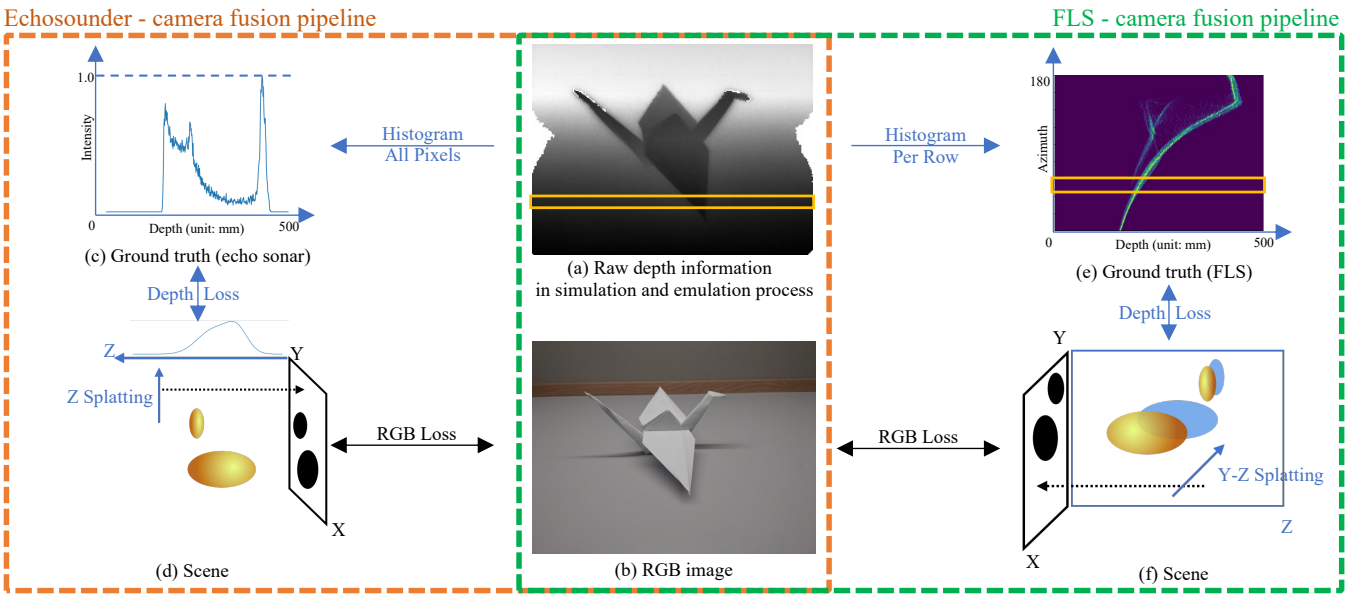


Fig. 3. **Simulation and emulation training for both echosounder and FLS fusion techniques** (a) Raw depth image captured with Time-of-Flight (ToF) camera. (b) An RGB image captured with a camera. (c) Simulated echosounder intensity was generated using the depth histogram and utilized as ground truth during training. (d) A 3D Gaussian scene. We use xy-splatting to render RGB images and z-splatting to render echosounder depth intensity distribution. (e) Simulated FLS intensity generated by histogramming depth per row. (f) A 3D Gaussian scene, and we splat along xy-direction to render RGB image and along yz-direction to render FLS image. We minimize the sum of RGB loss and corresponding depth loss to train the camera-sonar fusion algorithms.

5 Experimental Results

In this section, we systematically evaluate the proposed fusion techniques using simulated, emulated, and hardware-capture data. Novel view synthesis refers to generating images from camera views not in the training data. Using standard photometric metrics (PSNR, SSIM, and LPIPS), we compare the novel views generated by traditional Gaussian splatting and the proposed fusion techniques. We also compare the reconstructed geometry as 3D point clouds using Chamfer distance, F1 score, precision, and recall.

5.1 Simulation results

We use Mitsuba [50] to generate the simulated datasets. For each camera and sonar view, we render the image from the camera view and the depth map from the sonar view. We simulate echosounder intensity by histogramming the depth values of the entire scene, resulting in an intensity distribution across depth that mimics the transient sonar measurements. This histogram serves as the ground truth either for training or for testing. The left part of Figure 3 illustrates the process of generating the ground truth echosounder intensity and the predicted echosounder intensity using the proposed method.

To simulate FLS data, we histogram the depth values per row rather than the whole image. Each row represents one azimuth angle θ in the FLS. The resolution of the histogram along the z-axis is determined by the range r that the FLS can resolve. The right part of Figure 3 illustrates the process of generating ground-truth FLS scans. To initialize Gaussians, we found that random initialization to be more reliable than COLMAP initialization, as the latter sometimes suffered due to small baselines in our datasets.

We provide two different types of scenes in the simulation process: (i) Room-sized scenes from Mitsuba Gallery (Figure 4) and (ii) Single-object scenes from Mitsuba Gallery, Stanford 3D Scanning Repository and online free resources. In the room-sized scenes shown in Figure 4, we cannot freely maneuver the camera around the scene or object and obtain a full 360-degree scan due to the geometric constraints, and hence, we are in a restricted baseline scenario. Further, the scenes contain saturated regions due to windows and little texture. For these imaging scenarios, the camera-based reconstruction is ill-posed, and the addition of depth information better constrains the reconstruction problem.

We present the novel view synthesis comparisons (photometric comparisons) in Table 2 and geometry reconstruction comparisons in Table 1. For novel view synthesis, we outperform the RGB-only method on PSNR, SSIM, and LPIPS metrics. Specifically, we observe an average 5 dB improvement in PSNR compared to camera-only GS algorithms and a 10 dB increase on challenging scenes that contain little texture (living room scene).

The reconstructed 3D Gaussians typically have a small variance (of a couple of pixels) and, hence, can act as reconstructed point clouds. We tessellated the ground truth mesh and considered the mesh vertices as the ground truth point cloud. This tessellation allows us to compute various metrics (Chamfer distance, Recall, precision, F1-score) between the reconstructed point cloud and the ground truth. In Table 1, we show the comparisons across all the methods. The proposed techniques consistently outperform the traditional GS algorithms. we observe an average 50% improvement in Chamfer distance and a tenfold increase in the scenes with little texture (living room scene).

The primary reason for these improvements lies in our method's effective mitigation of the issue of floaters in the scene, a common challenge encountered in volumetric rendering methods. By reducing erroneous placement of Gaussians and enhancing both geometry accuracy and photometric details, we achieve significant advancements

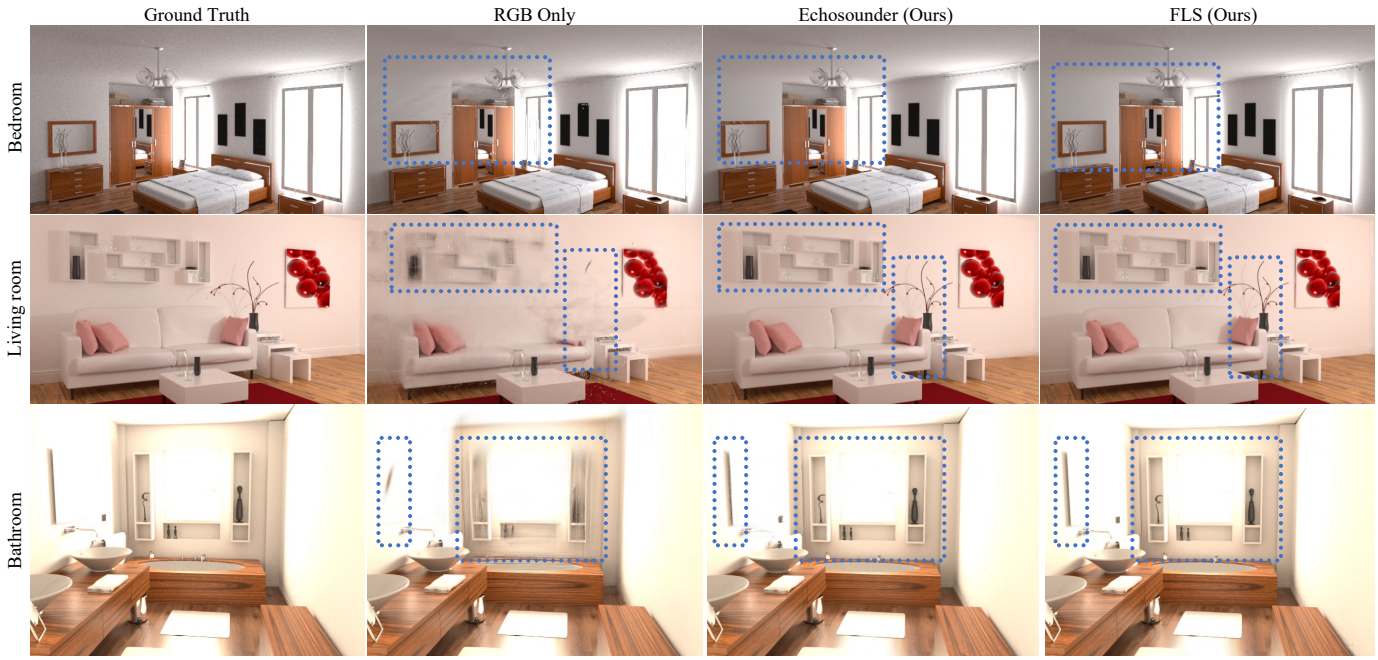


Fig. 4. **Novel view synthesis comparison:** The incorporation of depth information notably mitigates the presence of floaters in the reconstructed scene. Moreover, depth information accurately positions the Gaussian kernels, particularly in scenes with uniform color or overexposure. The average SSIM, PSNR, and LPIPS metrics for the entire test set comprising 263 novel views are presented in Table 1.

in overall performance. The results also indicate that FLS outperforms echosounder, albeit with only a slight improvement. This confirms our hypothesis from Section 4.2 that FLS will provide little additional information in the y-axis, as cameras also provide information along that dimension and at a higher resolution.

In the second set of scenes, instead of complex scenes with multiple objects, we created a series of scenes with only one diffuse object without texture. Similar to Qadri et al. [21], we restricted camera and sonar movement along the x-axis within a small range (2.3 degree on the circle) across these simplified scenes. We visualize the results in Figure 5, showcasing the ground truth as a mesh and the predicted Gaussian kernels as red points. In Table 3, we quantify the accuracy of reconstructed geometry with various metrics. We can observe that the proposed fusion consistently performs better than camera-only techniques.

Comparison to ToRF [47]. We found that the camera-only GS method outperforms ToRF. GS achieves a PSNR of 33.91 dB when trained solely on RGB images from the ToRF bedroom dataset, and 36.42 dB with fusion, compared to ToRF's PSNR of 29.79 dB with fusion. Note that the PSNR for ToRF is taken from its original paper.

5.2 Hardware emulation results

In addition to the simulation, we emulate the sonars using the camera and Lidar on the Sony Xperia II smartphone. We have built a Cornell box scene with multiple objects in our lab. We translate the phone inch-by-inch on the xy-plane to obtain training data. We capture RGB images and pixel-wise depth measurements simultaneously, mirroring the process employed during simulation in Section 5.1. We position the phone in between the training samples to generate the test dataset. We extract camera poses and Structure-from-Motion (SFM) points using COLMAP and train the three models of GS, Echosounder, and FLS.

The test results are shown in Figure 6 and quantified in Table 4. From Figure 6, it is evident that the RGB-only reconstruction results in significantly more blur on the white crane, box edges, and egg doll. This could be attributed to COLMAP creating denser points on clear edges but not on the background or white objects that share the same color as the background. During optimization, the RGB-only method would not produce more Gaussians because of the similar color. However, our methods surpass the RGB-only approach by producing clear and sharp edges with the aid of z-axis information. The average PSNR of our methods is 5 dB higher than that of RGB-only, and our techniques also have higher SSIM and lower LPIPS, as shown in Table 4.

During the experiments, we observed that the RGB-only method exhibits high variability. To investigate further, we conducted 10 training runs for each model using the same training dataset, test dataset, and initialized SFM points, but with random initial seeds. In Figure 7, we show the box and whisker plot to visualize the variance between the runs. The RGB-only method can perform poorly at times and exhibits significant variance across each training process. In contrast, both of our methods are robust and have consistent performance. Furthermore, even our worst-performing run outperforms the RGB-only method.

5.3 Hardware results - Echosounder

For our set of real hardware experiments, we first utilize our proposed method for sensor fusion between RGB cameras and a monostatic active sonar to capture acoustic ToF data. Echo sounding is the technique of using an active sonar to determine depth/range. Thus we have built a prototype of an echosounder using a single speaker and microphone that is attached to a DSLR camera for collecting real acoustic ToF data along with RGB images. We describe our setup and data processing procedures below.

Experimental setup and data capture: Our setup, visualized in Figure 8 and similar to other circular in-air

TABLE 1
Simulation: Novel view synthesis comparisons (Room sized scenes)

	RGB Only			Echosounder (Ours)			FLS (Ours)		
Scene	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Bedroom	31.855	0.881	0.242	35.264	0.893	0.205	35.348	0.891	0.177
Living room	27.508	0.874	0.331	37.790	0.948	0.182	38.457	0.951	0.176
Bathroom	27.465	0.872	0.177	33.381	0.929	0.092	35.753	0.952	0.073

PSNR, SSIM, and LPIPS metrics. We use PSNR, SSIM, and LPIPS to evaluate the quality of the predicted novel view RGB images. The best results are in green, while the worst are in red. We can see that our methods outperform the RGB-only method in all scenes. While the FLS method has the best performance, as expected, the improvement compared to the echosounder is small.

TABLE 2
Simulation: Geometric comparisons (Room sized scenes)

	RGB Only				Echosounder (Ours)				FLS (Ours)			
Scene	Chamfer↓	Precision↑	Recall↑	F1↑	Chamfer↓	Precision↑	Recall↑	F1↑	Chamfer↓	Precision↑	Recall↑	F1↑
Bedroom	0.374	0.893	0.549	0.680	0.163	0.997	0.667	0.799	0.198	0.998	0.622	0.767
Living Room	3.382	0.825	0.084	0.152	0.291	0.977	0.512	0.672	0.359	0.998	0.540	0.701
Bathroom	4.545	0.616	0.594	0.605	4.136	0.882	0.521	0.655	3.912	0.907	0.485	0.632

Chamfer, Precision, Recall, and F1 score metrics. We use Chamfer, Precision, Recall, and F1 score to evaluate the quality of the predicted 3D Gaussian splatting.The best results are in green, while the worst are in red. The metrics are calculated by comparing the point cloud of the predicted Gaussian splatting with the ground truth mesh vertices. Our methods outperform the RGB only method in most metrics, and the FLS method has the best performance.

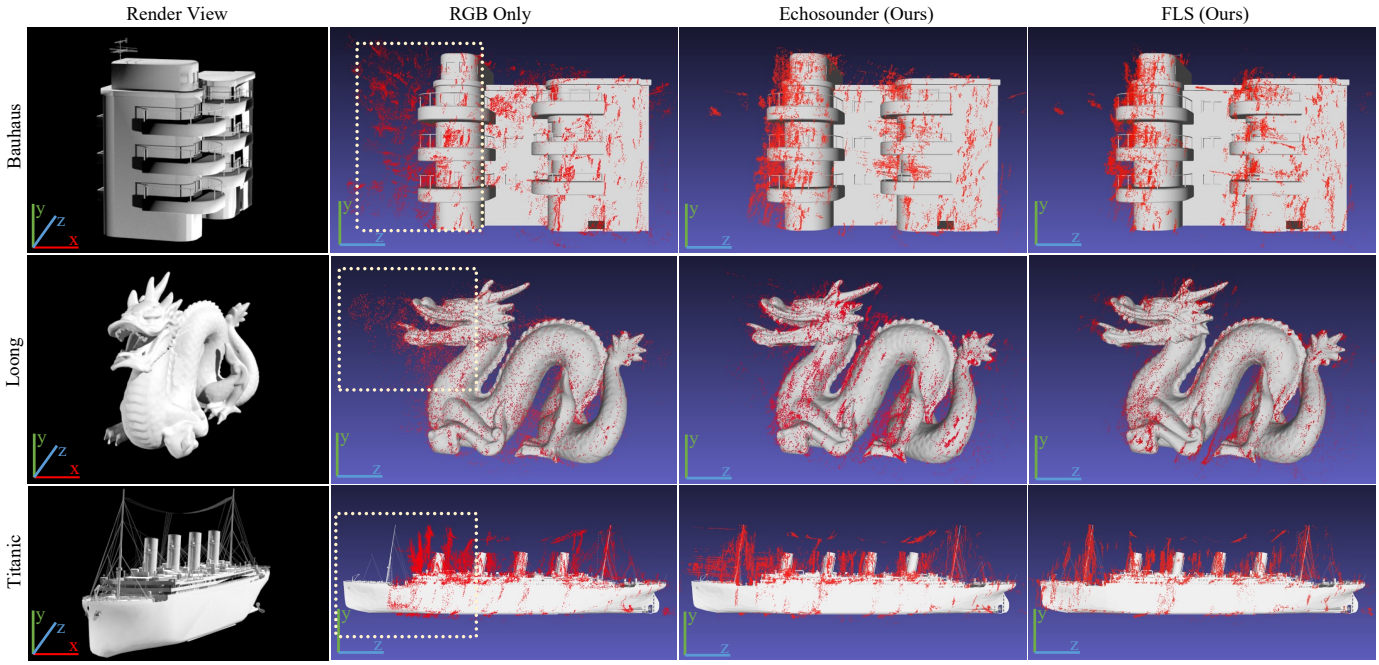


Fig. 5. **Geometry comparison on one-object scenes.** We captured the data by moving the camera only along the x-axis. We show ground truth meshes and superimpose the reconstructed Gaussians as point clouds. In the highlighted regions, we can observe that camera-only methods reconstruct the geometry inaccurately along the z-axis, whereas the proposed fusion techniques reconstruct the geometry accurately.

TABLE 3
Simulation: Geometric comparisons (One-object scenes)

	RGB Only				Echosounder (Ours)				FLS (Ours)			
Scene	Chamfer↓	Precision↑	Recall↑	F1↑	Chamfer↓	Precision↑	Recall↑	F1↑	Chamfer↓	Precision↑	Recall↑	F1↑
Bauhaus	0.184	0.943	0.693	0.799	0.134	0.899	0.729	0.805	0.127	0.894	0.772	0.829
Loong	0.0184	0.8462	0.916	0.846	0.0093	0.8627	0.944	0.902	0.0086	0.8815	0.938	0.909
Titanic	0.164	0.949	0.808	0.873	0.103	0.997	0.887	0.939	0.069	0.984	0.932	0.957

Chamfer, Precision, Recall, and F1 score metrics. We use Chamfer, Precision, Recall, and F1 score to evaluate the quality of the predicted 3D Gaussian splatting. The best results are in green, while the worst are in red. These metrics are computed by comparing the point cloud of the predicted Gaussian splatting with the ground truth mesh vertices. Our methods consistently outperform the RGB-only approach across most metrics, and FLS exhibits slightly better performance than echosounder.

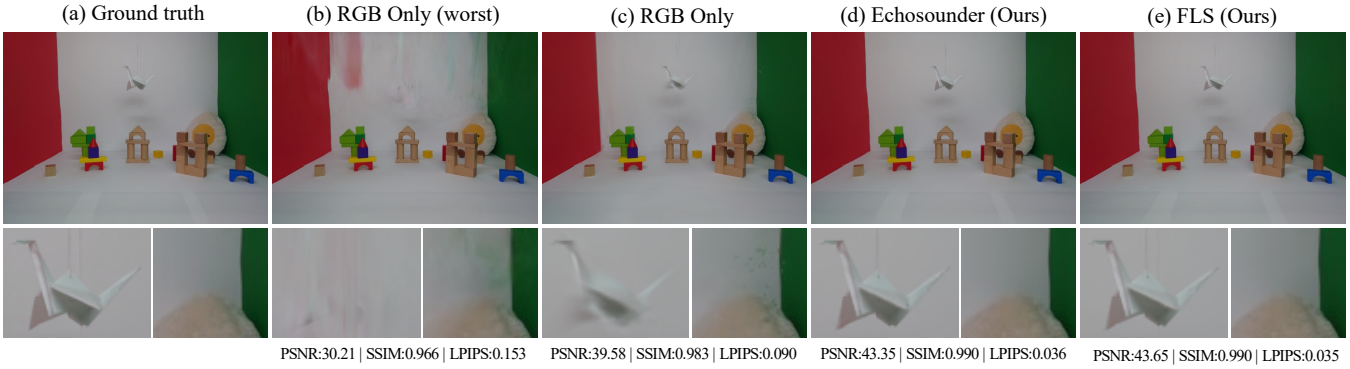


Fig. 6. **Novel view synthesis comparison on emulated hardware.** We set up a Cornell box in the lab, captured both RGB and depth images, and emulated the echosounder and FLS data. In the reconstructed scene, all methods work well on the objects with high-contrast textures. However, the RGB-only technique fails to reconstruct the white object with the same color as the background and also suffers from color bleeding. Our methods, on the other hand, successfully reconstruct the white object and do not suffer from color bleeding. For different random seeds, RGB-only techniques have high variance in the reconstructed results and have poor reconstructions (b), while our methods have consistent performance. We plot the variance across different runs in Figure 7, and we can observe that our techniques are robust to random initial seeds.

TABLE 4
Emulation Performance: Cornell Box

	PSNR↑	SSIM↑	LPIPS↓
RGB Only	37.499	0.977	0.093
Echosounder (Ours)	42.089	0.987	0.037
FLS (Ours)	**42.142**	**0.988**	**0.036**

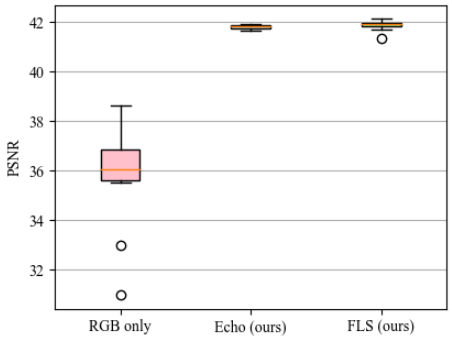


Fig. 7. **Variance comparison for emulations.** We ran the GS algorithms 10 times with the same training and test datasets but with different random seeds. We can see that the variance of our methods is much lower than the RGB-only method. This indicates that our methods have consistent performance, while the RGB-only method can sometimes result in poor results. Note that even the best results with RGB-only are worse than the proposed techniques.

systems [51–53], consists of (1) acoustic transducer array consisting of loudspeaker tweeter (Peerless by Tymphany OX20SC02-04) and a microphone (GRAS 46AM); (2) a Nikon D5600 DSLR camera; and (3) a motorized circular platform (Rotary Table RTLA-90-200M) to rotate targets. As seen in the figure, white paper and a white board was used to cover the turntable and background, otherwise parts of the scene would be static while the objects moved, not reflecting the experimental scenario of a moving sonar. The tweeter transmits a linear frequency modulated chirp with peak voltage of 3V for a duration of 1 ms waveform from starting frequency $10kHz$ to stopping frequency $30kHz$ with a sampling frequency of $100kHz$.

For data capture, we rotate the target scene at 1-degree increments from 0 to 40 degrees while simultaneously capturing RGB video frames and the acoustic signal from the

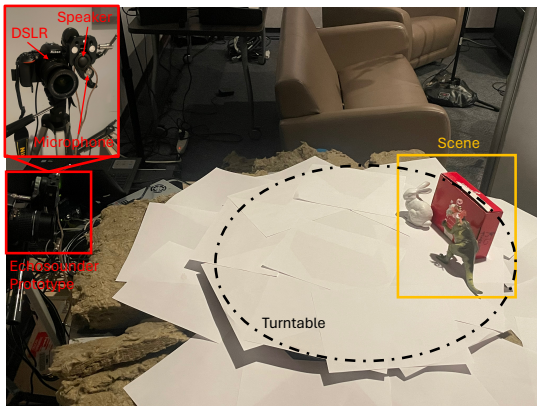


Fig. 8. **Hardware prototype** consists of a DSLR camera with speaker and microphone. It is used to image a scene on the motorized turntable.

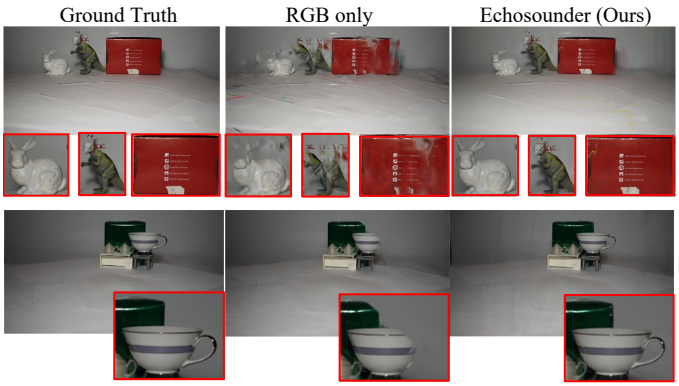


Fig. 9. **Qualitative comparison of echosounder real-data results.** The comparison presents two scenes captured using a DSLR camera and echo-sonar with a turntable setup. Our method demonstrates a noticeable improvement in performance over the baseline RGB-only method, both quantitatively and qualitatively.

microphone. We first use COLMAP to perform SfM to obtain an initial point cloud and camera pose. We perform model alignment for SFM points using the information from the experimental setup (radius of the turn table and incremental angle) to roughly align the SFM points in 3D space with the transient histogram to be computed. For the audio signal, we perform similar processing for matched filtering by Reed et al. [22, 53]: (1) we implement group delay correction in the frequency domain; (2) we subtract background measure-

TABLE 5
Echo-Sonar Performance

	RGB Only			Echosounder (Ours)		
Scene	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Rabbit	25.882	0.874	0.282	**30.785**	**0.940**	**0.237**
Teapot	39.664	0.982	0.138	**39.717**	**0.985**	**0.114**

ment of the scene with no object; (3) we convert the signal into the analytic domain using the Hilbert Transform; (3) we perform matched filtering in the frequency domain, and then (4) take the magnitude of the complex signal to form a transient histogram.

For initialization of the Gaussians, we utilize the inital SfM point cloud generated from COLMAP as well as include additional 3D Gaussians ($N = 10,000$ for our experiments) who are placed at (x, y, z) locations corresponding to maximum energy in the transient histogram. This greatly improved the performance of the method in practice compared to the baseline Gaussian splatting.

Main experimental results: In Fig. 9, we show a qualitative comparison of our method compared to an RGB only 3D reconstruction. This scene is a difficult scene as COLMAP SfM points are noisy and inaccurate, which contributes to errors in the RGB only Gaussian-splatting reconstruction. In contrast, our method better resolves the objects in the scene as well as the background, leveraging the transient information from the echosounder. In Tab. 5, we see an improvement in PSNR/SSIM/LPIPS for test views.

Effect of baseline: We designed the technique for small baselines (small shifts in the camera positions) where sonar provides the most information in the missing cone (Section 3.4) and showed that the proposed technique results in higher 3D reconstruction accuracy compared to RGB case. As the baseline increases, the missing cone becomes narrower; hence, we will only have diminishing gains from the sonar. Here, we quantify the effect of the baseline on the reconstruction performance. For this, we rotate the circular table by various ranges of angles and compare the geometric reconstructions between RGB only and the proposed fusion techniques. We show the visual reconstruction comparisons in Figure 10 and quantify them in Table 6. We observe that fusion techniques have a significant advantage at small baselines (20°), achieving a 24.5% lower Chamfer distance, but a relatively lower advantage at large baselines (180°), with only a 7.9% decrease in Chamfer distance.

TABLE 6
Various Baselines Results

Baseline	Metrics	RGB only	Ours echosounder
20°	Chamfer ↓	0.01455	0.01098
	F1 ↑	0.9534	0.9910
40°	Chamfer ↓	0.00845	0.00656
	F1 ↑	0.9903	0.9962
90°	Chamfer ↓	0.0042	0.0039
	F1 ↑	0.9985	0.9989
180°	Chamfer ↓	0.00366	0.00337
	F1 ↑	0.9994	0.9997

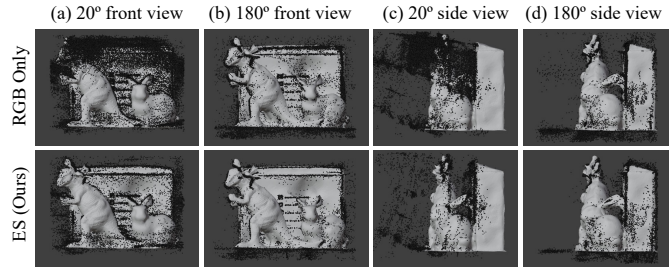


Fig. 10. **Effect of baseline.** We visualize two extreme baselines: $20°$ and $180°$. The ground truth meshes captured with LIDAR are gray, and the reconstructed Gaussians are represented as black point clouds. In both baselines, our method demonstrates better alignment to the underlying geometry and less error in the z axis, resulting in fewer Gaussians and clearer representations. As the baseline increases, our method's advantage over the RGB-only approach diminishes but still remains positive.

5.4 Hardware results - FLS

We additionally use our algorithm to fuse measurements from an optical camera and an imaging sonar (FLS) to reconstruct a real object submerged underwater in a water test tank. Acoustic FLS images (example in fig.11a) resolve both the range and azimuth of the reflecting object but not the elevation angle which remains ambiguous. In other words, one can view each column of an imaging sonar image as capturing the transient at a particular azimuth angle which makes this data easily integrable with our method.

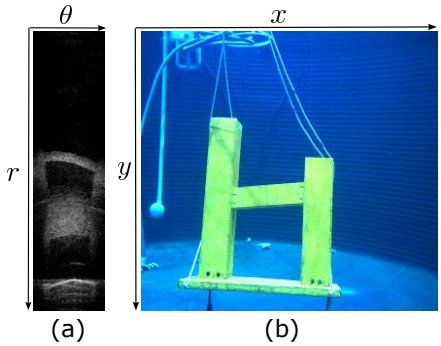


Fig. 11. **Underwater FLS data.** (a) Example sonar measurement: the range r and azimuth θ are resolved but the elevation angle ϕ remains ambiguous. (b) Sample camera image of the submerged test structure.

We use an existing dataset containing both imaging sonar measurements (with $14°$ elevation) captured using a Didson imaging sonar mounted on a Bluefin Autonomous Underwater Vehicle and optical camera images captured using a FLIR camera. See papers [21, 54] for more detail regarding the data and hardware setup. Note that for training, we use all available measurements in the dataset while AONeuS [21] only uses specific subsamples.

Due to the scattering in water, GS is not suitable for underwater photometric reconstruction, which could be addressed in future work. However, we compare geometric reconstructions. For this, we first filter out all points outside a bounding box around the object. Then, we align the resulting point cloud with the ground truth model of the target object using the Iterative Closest Point (ICP) algorithm.

We observe from Table 7 that integrating FLS information with camera images results in higher reconstruction accuracy, as showcased by the Chamfer distance/precision/recall/F1 metrics (threshold 0.05). This can

TABLE 7
Hardware FLS: Geometric Metrics

	Chamfer ↓	Precision↑	Recall↑	F1↑
RGB Only	0.205	0.414	0.670	0.512
FLS (Ours)	**0.124**	**0.457**	**0.775**	**0.575**

also be observed quantitatively in Figure 12, which shows that when only using optical cameras, the resulting reconstruction contains high errors along the depth axis. On the other hand, integrating FLS information allows our algorithm to better resolve depth using the additional range information. We note that, although our result is slightly worse compared to AONeuS [21] (although do expect better performance with more thorough parameter tuning), the runtime of our algorithm is $\sim 9.95\times$ better.

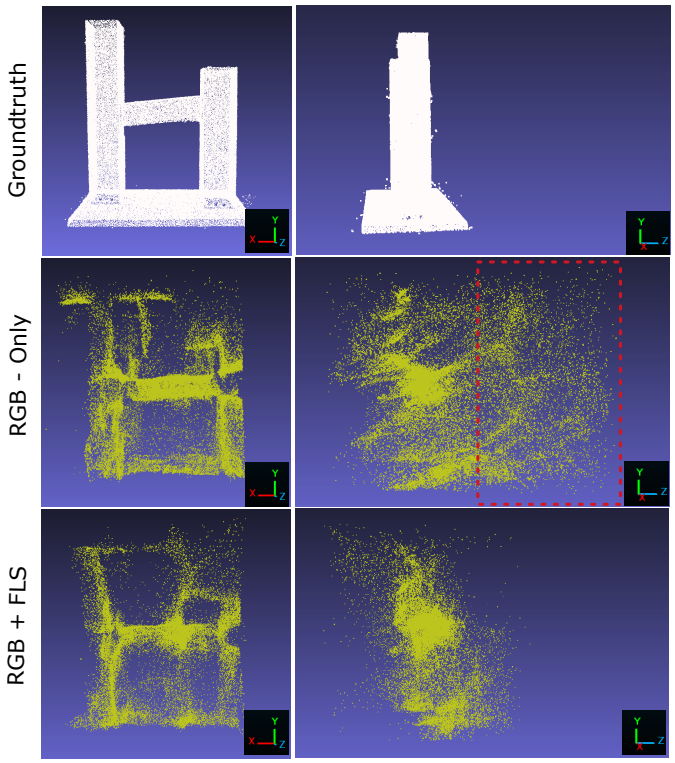


Fig. 12. **Experimental reconstructions using RGB-only and RGB+FLS.** When using RGB-only measurements, we observe high error along depth (red box). Integrating FLS measurements improves depth resolution.

6 Conclusion

In this paper, we introduce a z-axis splatting pipeline for Gaussian splatting, which enables the fusion of RGB camera information with various types of depth-resolved acoustic measurements, such as echo-sounders and FLS. Through extensive validation on simulated, emulated, and real-world hardware experimental data, we demonstrate superior performance compared to RGB-only methods for both novel view synthesis and geometric reconstruction. By leveraging the depth information provided by acoustic sensors, our method offers a promising approach to reconstructing scenes with small baselines.

Our method currently does not account for scattering. Modeling scattering may produce more accurate reconstructions, particularly in the underwater setting. In the fusion step, we used a simple linear model to combine the losses of various imaging modalities. Non-linear models and adaptive models that change the relative weights over iterations, such as the ones that Qadri et al. [21] have used, may result in better 3D reconstruction. Exploring various combinations of loss functions could be another interesting future direction.

While this work focused on z-axis splatting for sonar fusion, an interesting future direction is to extend our approach to handle radar and lidar systems, which share a similar acquisition model as sonars. Finally, generalizing our method to dynamic scenes and to sensors that operate in this space, such as Doppler cameras and frequency-modulated continuous-wave time-of-flight cameras, could be another interesting extension of our technique.

Acknowledgments

A.P. and Z.Q. were supported in part by NSF CCF-2326904. O.V. and S.J. were supported in part by ONR grant N00014-23-1-2406, NSF CCF-2326905, and Raytheon, Inc. In addition, the authors acknowledge Research Computing at Arizona State University for providing the Sol supercomputer [55] for GPU computing. M.Q. and M.K. were supported in part by ONR grant N00014-21-1-2482. K.Z. and C.A.M. were supported in part by AFOSR Young Investigator Program award no. FA9550-22-1-0208, ONR award no. N00014-23-1-2752, and a seed grant from SAAB, Inc.

References

[1] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," ACM Transactions on Graphics, vol. 42, no. 4, pp. 1–14, 2023.
[2] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross, "Ewa volume splatting," in Proceedings Visualization, 2001. VIS'01. IEEE, 2001, pp. 29–538.
[3] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai, "Scaffold-gs: Structured 3d gaussians for view-adaptive rendering," arXiv preprint arXiv:2312.00109, 2023.
[4] Z. Yu, A. Chen, B. Huang, T. Sattler, and A. Geiger, "Mip-splatting: Alias-free 3d gaussian splatting," arXiv preprint arXiv:2311.16493, 2023.
[5] 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," Communications of the ACM, vol. 65, no. 1, pp. 99–106, 2021.
[6] J. Gao, C. Gu, Y. Lin, H. Zhu, X. Cao, L. Zhang, and Y. Yao, "Relightable 3d gaussian: Real-time point cloud relighting with brdf decomposition and ray tracing," arXiv preprint arXiv:2311.16043, 2023.
[7] Y. Jiang, J. Tu, Y. Liu, X. Gao, X. Long, W. Wang, and Y. Ma, "Gaussianshader: 3d gaussian splatting with shading functions for reflective surfaces," arXiv preprint arXiv:2311.17977, 2023.
[8] Z. Zhu, Z. Fan, Y. Jiang, and Z. Wang, "Fsgs: Real-time few-shot view synthesis using gaussian splatting," arXiv preprint arXiv:2312.00451, 2023.
[9] J. Fang, J. Wang, X. Zhang, L. Xie, and Q. Tian, "Gaussianeditor: Editing 3d gaussians delicately with text instructions," arXiv preprint arXiv:2311.16037, 2023.
[10] D. Das, C. Wewer, R. Yunus, E. Ilg, and J. E. Lenssen, "Neural parametric gaussians for monocular non-rigid object reconstruction," arXiv preprint arXiv:2312.01196, 2023.
[11] Z. Yang, H. Yang, Z. Pan, X. Zhu, and L. Zhang, "Real-time photorealistic dynamic scene representation and

rendering with 4d gaussian splatting," arXiv preprint arXiv:2310.10642, 2023.
[12] A. Kratimenos, J. Lei, and K. Daniilidis, "Dynmf: Neural motion factorization for real-time dynamic view synthesis with 3d gaussian splatting," arXiv preprint arXiv:2312.00112, 2023.
[13] R. Shao, J. Sun, C. Peng, Z. Zheng, B. Zhou, H. Zhang, and Y. Liu, "Control4d: Dynamic portrait editing by learning 4d gan from 2d diffusion-based editor," arXiv preprint arXiv:2305.20082, 2023.
[14] Y.-H. Huang, Y.-T. Sun, Z. Yang, X. Lyu, Y.-P. Cao, and X. Qi, "Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes," arXiv preprint arXiv:2312.14937, 2023.
[15] H. Yu, J. Julin, Z. Á. Milacski, K. Niinuma, and L. A. Jeni, "Cogs: Controllable gaussian splatting," arXiv preprint arXiv:2312.05664, 2023.
[16] J. Huang and H. Yu, "Point'n move: Interactive scene object manipulation on gaussian splatting radiance fields," arXiv preprint arXiv:2311.16737, 2023.
[17] F. Macias-Garza, K. R. Diller, and A. C. Bovik, "Missing cone of frequencies and low-pass distortion in three-dimensional microscopic images," Optical Engineering, vol. 27, no. 6, pp. 461–465, 1988.
[18] X. Liu, S. Bauer, and A. Velten, "Analysis of feature visibility in non-line-of-sight measurements," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 10 140–10 148.
[19] J. Chung, J. Oh, and K. M. Lee, "Depth-regularized optimization for 3d gaussian splatting in few-shot images," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 811–820.
[20] H. Liu, M. Roznere, and A. Q. Li, "Deep underwater monocular depth estimation with single-beam echosounder," in 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 1090–1097.
[21] M. Qadri, K. Zhang, A. Hinduja, M. Kaess, A. Pediredla, and C. A. Metzler, "Aoneus: A neural rendering framework for acoustic-optical sensor fusion," arXiv preprint arXiv:2402.03309, 2024.
[22] A. Reed, J. Kim, T. Blanford, A. Pediredla, D. Brown, and S. Jayasuriya, "Neural volumetric reconstruction for coherent synthetic aperture sonar," ACM Transactions on Graphics (TOG), vol. 42, no. 4, pp. 1–20, 2023.
[23] M. F. Fallon, J. Folkesson, H. McClelland, and J. J. Leonard, "Relocating underwater features autonomously using sonar-based slam," IEEE Journal of Oceanic Engineering, vol. 38, no. 3, pp. 500–513, 2013.
[24] P. Yang, H. Liu, M. Roznere, and A. Q. Li, "Monocular camera and single-beam sonar-based underwater collision-free navigation with domain randomization," in The International Symposium of Robotics Research. Springer, 2022, pp. 85–101.
[25] T. Lin, A. Hinduja, M. Qadri, and M. Kaess, "Conditional gans for sonar image filtering with applications to underwater occupancy mapping," in 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 1048–1054.
[26] M. Roznere and A. Q. Li, "Underwater monocular image depth estimation using single-beam echosounder," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2020, pp. 1785–1790.
[27] T. Zhang, S. Liu, X. He, H. Huang, and K. Hao, "Underwater target tracking using forward-looking sonar for autonomous underwater vehicles," Sensors, vol. 20, no. 1, p. 102, 2019.
[28] Z-splat: Python implementation. [Online]. Available: https://github.com/QuintonQu/gaussian-splatting-with-depth/tree/gs-depth-main
[29] P. S. Heckbert, "Fundamentals of texture mapping and image warping," 1989.
[30] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross, "Surface splatting," in Proceedings of the 28th annual conference on Computer graphics and interactive techniques, 2001, pp. 371–378.
[31] ——, "Ewa splatting," IEEE Transactions on Visualization and Computer Graphics, vol. 8, no. 3, pp. 223–238, 2002.
[32] B. Fei, J. Xu, R. Zhang, Q. Zhou, W. Yang, and Y. He, "3d gaussian as a new vision era: A survey," arXiv preprint arXiv:2402.07181, 2024.
[33] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting slam," arXiv preprint arXiv:2312.06741, 2023.
[34] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "Splatam: Splat, track & map 3d gaussians for dense rgb-d slam," arXiv preprint arXiv:2312.02126, 2023.
[35] C. Yan, D. Qu, D. Wang, D. Xu, Z. Wang, B. Zhao, and X. Li, "Gs-slam: Dense visual slam with 3d gaussian splatting," arXiv preprint arXiv:2311.11700, 2023.
[36] S. Sun, M. Mielle, A. J. Lilienthal, and M. Magnusson, "High-fidelity slam using gaussian splatting with rendering-guided densification and regularized optimization," arXiv preprint arXiv:2403.12535, 2024.
[37] F. Ferreira, D. Machado, G. Ferri, S. Dugelay, and J. Potter, "Underwater optical and acoustic imaging: A time for fusion? a brief overview of the state-of-the-art," OCEANS 2016 MTS/IEEE Monterey, pp. 1–6, 2016.
[38] Y. Raaj, A. John, and T. Jin, "3d object localization using forward looking sonar (fls) and optical camera via particle filter based calibration and fusion," OCEANS 2016 MTS/IEEE Monterey, pp. 1–10, 2016.
[39] S. Williams and I. Mahon, "Simultaneous localisation and mapping on the great barrier reef," IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004, vol. 2, pp. 1771–1776, 2004.
[40] M. Babaee and S. Negahdaripour, "3-d object modeling from 2-d occluding contour correspondences by opti-acoustic stereo imaging," Computer Vision and Image Understanding, vol. 132, pp. 56–74, 2015.
[41] R. Nabati and H. Qi, "Centerfusion: Center-based radar and camera fusion for 3d object detection," in Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2021, pp. 1527–1536.
[42] R. Grover, G. Brooker, and H. F. Durrant-Whyte, "A low level fusion of millimeter wave radar and night-vision imaging for enhanced characterization of a cluttered environment," in Proceedings 2001 Australian Conference on Robotics and Automation, 2001.
[43] M. Bijelic, T. Gruber, F. Mannan, F. Kraus, W. Ritter, K. Dietmayer, and F. Heide, "Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11 682–11 692.
[44] Y. M. Kim, C. Theobalt, J. Diebel, J. Kosecka, B. Miscusik, and S. Thrun, "Multi-view image and tof sensor fusion for dense 3d reconstruction," in 2009 IEEE 12th international conference on computer vision workshops, ICCV workshops. IEEE, 2009, pp. 1542–1549.
[45] D. B. Lindell, M. O'Toole, and G. Wetzstein, "Single-photon 3d imaging with deep sensor fusion." ACM Trans. Graph., vol. 37, no. 4, p. 113, 2018.
[46] M. Nishimura, D. B. Lindell, C. Metzler, and G. Wetzstein, "Disambiguating monocular depth estimation with a single transient," in European Conference on Computer Vision. Springer, 2020, pp. 139–155.
[47] B. Attal, E. Laidlaw, A. Gokaslan, C. Kim, C. Richardt,

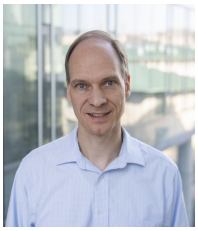
J. Tompkin, and M. O'Toole, "Törf: Time-of-flight radiance fields for dynamic scene view synthesis," Advances in neural information processing systems, vol. 34, pp. 26 289–26 301, 2021.
[48] M. Salvi and K. Vaidyanathan, "Multi-layer alpha blending," in Proceedings of the 18th Meeting of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games, 2014, pp. 151–158.
[49] A. H. Delaney and Y. Bresler, "Globally convergent edge-preserving regularized reconstruction: an application to limited-angle tomography," IEEE Transactions on image processing, vol. 7, no. 2, pp. 204–221, 1998.
[50] M. Nimier-David, D. Vicini, T. Zeltner, and W. Jakob, "Mitsuba 2: A retargetable forward and inverse renderer," ACM Transactions on Graphics (TOG), vol. 38, no. 6, pp. 1–17, 2019.
[51] T. E. Blanford, J. D. McKay, D. C. Brown, J. D. Park, and S. F. Johnson, "Development of an in-air circular synthetic aperture sonar system as an educational tool," vol. 36. ASA, 8 2019, p. 070002.
[52] J. D. Park, T. E. Blanford, D. C. Brown, and D. Plotnick, "Alternative representations and object classification of circular synthetic aperture in-air acoustic data," The Journal of the Acoustical Society of America, vol. 148, no. 4_Supplement, pp. 2661–2661, 2020.
[53] A. Reed, T. Blanford, D. C. Brown, and S. Jayasuriya, "Sinr: Deconvolving circular sas images using implicit neural representations," IEEE Journal of Selected Topics in Signal Processing, 2022.
[54] M. Qadri, M. Kaess, and I. Gkioulekas, "Neural implicit surface reconstruction using imaging sonar," in 2023 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2023, pp. 1040–1047.
[55] D. M. Jennewein, J. Lee, C. Kurtz, W. Dizon, I. Shaeffer, A. Chapman, A. Chiquete, J. Burks, A. Carlson, N. Mason, A. Kobwala, T. Jagadeesan, P. Barghav, T. Battelle, R. Belshe, D. McCaffrey, M. Brazil, C. Inumella, K. Kuznia, J. Buzinski, S. Dudley, D. Shah, G. Speyer, and J. Yalim, "The Sol Supercomputer at Arizona State University," in Practice and Experience in Advanced Research Computing, ser. PEARC '23. New York, NY, USA: Association for Computing Machinery, Jul 2023, pp. 296–301.

Ziyuan Qu received his B.Eng. in Software Engineering from Northeastern University, China in 2022, and his M.S. in Computer Science from Dartmouth College in 2024. He is currently pursuing a PhD in Computer Science at Dartmouth College under supervision of Prof. Adithya Pediredla, focusing on computational photography and rendering.

Omkar Vengurlekar received his B.Eng. in Electronics and Telecommunication Engineering from Pune University, India in 2020, and his M.S. in Robotics and Autonomous Systems from Arizona State University in 2024. He is currently pursuing a PhD in Computer Engineering at Arizona State University under the supervision of Prof. Suren Jayasuriya, focusing on computational vision, 3D Reconstruction, and machine learning.

Mohamad Qadri received his BS in Electrical Engineering from the University of Maryland, College Park in 2016, and his MS in Robotics from Carnegie Mellon University in 2021. He is currently pursuing a PhD in Robotics at Carnegie Mellon University with a focus on computer vision.

Kevin Zhang received the B.A. in Computer Science and Pure Mathematics from the University of California, Berkeley. He is currently a Computer Science Ph.D. student at the University of Maryland, College Park, based primarily in the Intelligent Sensing Lab, advised by Professor Christopher Metzler and Jia-bin Huang. His research focuses on 3D reconstruction from various sensing modalities.



Michael Kaess (Senior Member, IEEE) received the M.S. and Ph.D. degrees in computer science from the Georgia Institute of Technology, Atlanta, GA, USA, in 2002 and 2008, respectively. He is an Associate Professor in the Robotics Institute at Carnegie Mellon University, Pittsburgh, PA, USA. Before he was a postdoctoral associate and research scientist at the Computer Science and Artificial Intelligence Laboratory (CSAIL) at the Massachusetts Institute of Technology (MIT), Cambridge, MA, USA. He is currently the Director of the Robot Perception Lab, Carnegie Mellon University. His research interests include state estimation, localization, mapping, navigation, and autonomy. Dr. Kaess is a member of IEEE, AAAI, and ACM. He is the recipient of the inaugural Robotics: Science and Systems Test of Time Award 2020.

Christopher A. Metzler is an Assistant Professor in the Department of Computer Science at the University of Maryland College Park, where he leads the UMD Intelligent Sensing Laboratory. He is a member of the University of Maryland Institute for Advanced Computer Studies (UMIACS) and has a courtesy appointment in the Electrical and Computer Engineering Department. His research develops new systems and algorithms for solving problems in computational imaging and sensing, machine learning, and wireless communications. His work has received multiple best paper awards; he recently received NSF CAREER, AFOSR Young Investigator Program, and ARO Early Career Program awards; and he was an Intelligence Community Postdoctoral Research Fellow, an NSF Graduate Research Fellow, a DoD NDSEG Fellow, and a NASA Texas Space Grant Consortium Fellow.

Suren Jayasuriya (Senior Member, IEEE) is an assistant professor at Arizona State University, in the School of Arts, Media and Engineering (AME) and Electrical, Computer and Energy Engineering (ECEE) since 2018. Before this, he was a postdoctoral fellow at the Robotics Institute at Carnegie Mellon University from 2016-2017. He received his Ph.D. in electrical and computer engineering at Cornell University in 2017 and graduated from the University of Pittsburgh in 2012 with a B.S. in Mathematics (with departmental honors) and a B.A. in Philosophy. His research interests range from computational imaging and photography, computer vision and graphics, and machine learning.

Adithya Pediredla is an Assistant Professor in the Department of Computer Science at Dartmouth College, where he leads the Rendering and Imaging Science (RISC) lab, focusing on the intersection of computer graphics, computational imaging, and computer vision. Before this role, he is a postdoctoral fellow at the Robotics Institute, Carnegie Mellon University. He has a Ph.D. from Rice University and a master's degree from the Indian Institute of Science. He won the Ralph Budd Best Engineering Thesis Award for his Ph.D. thesis, the Prof. K. R. Kambati Memorial gold medal, and an innovative student project award from the Indian National Academy of Engineering for his Master's thesis.