Gaussian Splashing Enables Direct Volumetric Rendering Underwater

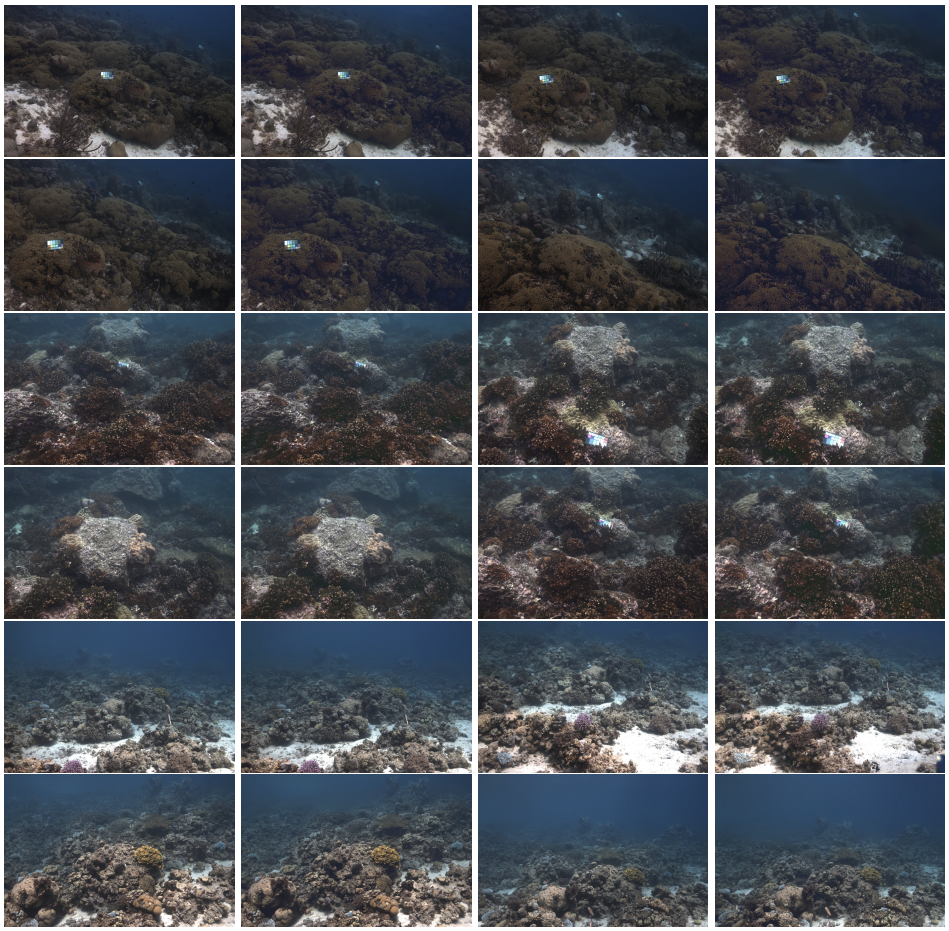
Nir Mualem, Roy Amoyal, Oren Freifeld, and Derya Akkaynak

Supplementary Information

A Supplemental Material

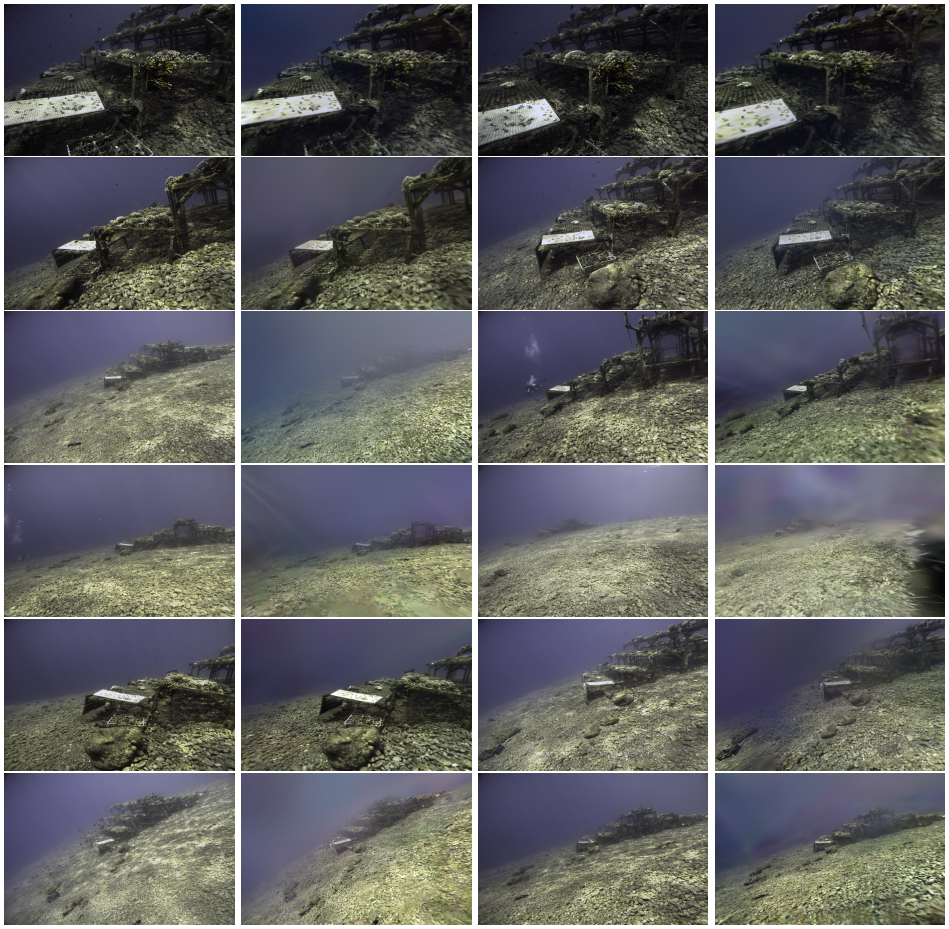
A.1 Additional Visual Results

Qualitative videos showcasing comparisons, depth estimation, and backscattering estimation, including videos for frames from our new TableDB dataset, are available on **our webpage**, and we strongly encourage viewing them to get a better impression and deeper understanding. Additional rendering results for training images and novel views are presented in Figure 1, where Figure 2 includes several images from our new TableDB dataset (more examples from TableDB are available in our webpage). Figure 3 provides additional visualizations of rendered views together with depth maps. Figure 4 shows an example of color reconstruction enabled by the depth maps produced by our method.



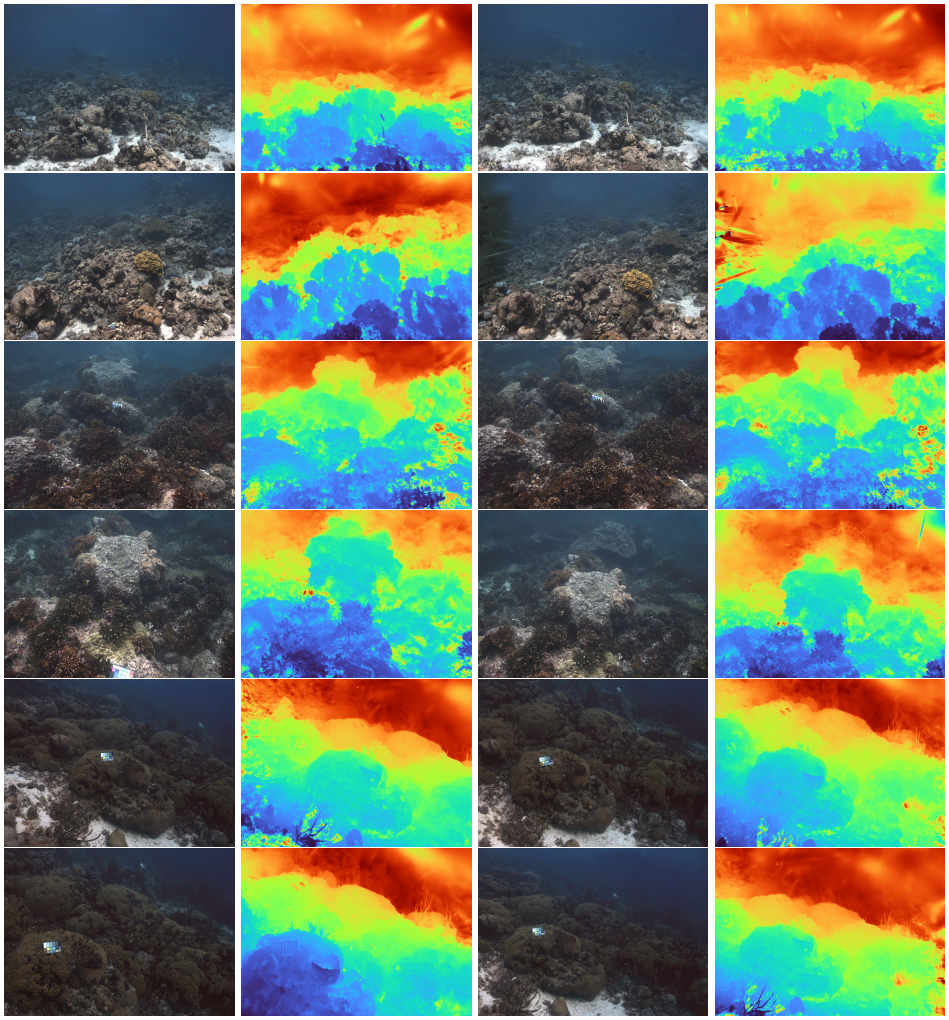
(a) Ground Truth (train) (b) Rendered (train) (c) Ground Truth (test) (d) Rendered (test)

Figure 1: Supplementary rendering results exemplifying the effectiveness and visual quality of our method across three different scenes and conditions. Train refers to the data used during the reconstruction. Test refers to novel views.



(a) Ground Truth (train) (b) Rendered (train) (c) Ground Truth (test) (d) Rendered (test)

Figure 2: Supplementary rendering results demonstrating the effectiveness and visual quality of our method on the TableDB dataset. TableDB is a distinctive underwater dataset featuring a wide range of depths. "Train" refers to data used for reconstruction, while "Test" represents novel views.



(a) Rendered (train) (b) Depth (train) (c) Rendered (test) (d) Depth (test)

Figure 3: Rendered views and novel views, along with the depth maps generated by our method, shown for scenes from the Red Sea, Panama, and Curaçao.

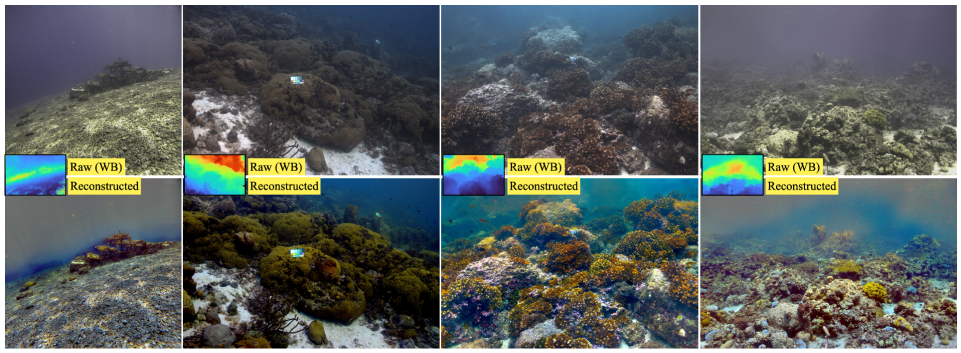


Figure 4: Color reconstruction application utilizing depth maps generated by our method, with raw (white-balanced, WB) images as inputs to the original Sea-thru algorithm. Results are presented across all datasets.

A.2 Partial Derivatives of the New Parameters

$$
\begin{aligned}
C_{\mathrm{UW}} &= \left[\sum_{i=1}^{N} \boldsymbol{c}_i \alpha_i \cdot T_i\right] e^{-B_{\mathrm{d}} z} + B_{\infty}(1 - e^{-B_{\mathrm{b}} z}) \\
&= \left[\sum_{i=1}^{N} \boldsymbol{c}_i \alpha_i \cdot \prod_{j=0}^{i-1}(1-\alpha_j)\right] e^{-B_{\mathrm{d}} z} + B_{\infty}(1 - e^{-B_{\mathrm{b}} z}) \qquad (1)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}_i} &= \frac{\partial C_{\mathrm{UW}}}{\partial \boldsymbol{c}_i} \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \\
&= \alpha_i T_i \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \\
&= \alpha_i \prod_{j=0}^{i-1}(1-\alpha_j) \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \qquad (2)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial \alpha_i} &= \frac{\partial C_{\mathrm{UW}}}{\partial \alpha_i} \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \\
&= \left[\boldsymbol{c}_i \cdot T_i - \sum_{j=i+1}^{N-1} \boldsymbol{c}_j \alpha_j \frac{T_j}{(1-\alpha_i)}\right] \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \\
&= \left[\boldsymbol{c}_i \prod_{j=0}^{i-1}(1-\alpha_j) - \sum_{j=i+1}^{N-1} \boldsymbol{c}_j \alpha_j \left(\prod_{k=0}^{i-1}(1-\alpha_k) \prod_{k=i+1}^{j-1}(1-\alpha_k)\right)\right] \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \qquad (3)
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial \mathcal{L}}{\partial B_{\mathrm{d}}} &= -\sum_{i=1}^{N} (\boldsymbol{c}_i \alpha_i T_i) \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \\
&= -\sum_{i=1}^{N} \left(\boldsymbol{c}_i \alpha_i \prod_{j=0}^{i-1}(1-\alpha_j)\right) \cdot e^{-B_{\mathrm{d}} z} \cdot \frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}} \qquad (4)
\end{aligned}
$$

$$
\frac{\partial \mathcal{L}}{\partial B_{\mathrm{b}}} = \left(\frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}}\right) B_{\infty} e^{-B_{\mathrm{b}} z} + \lambda_2 \qquad (5)
$$

$$
\frac{\partial \mathcal{L}}{\partial B_{\infty}} = \left(\frac{\partial \mathcal{L}}{\partial C_{\mathrm{UW}}}\right) (1 - e^{-B_{\mathrm{b}} z}) + \lambda_2 \qquad (6)
$$

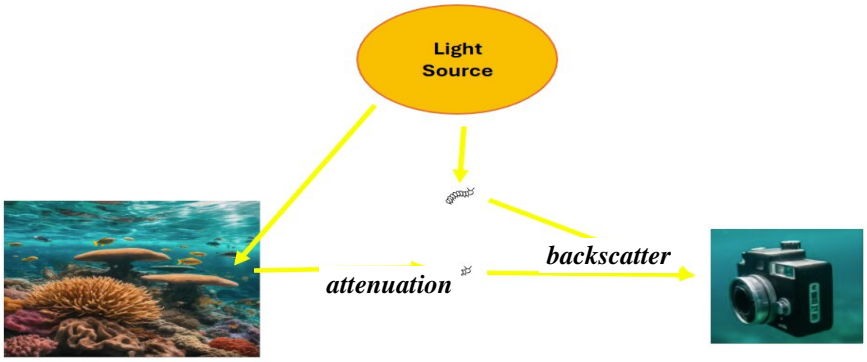


Figure 5: Illustration of the scattering and absorption of light by waterborne particles, which then reflect back to the camera. This phenomenon leads to image degradation by reducing contrast, altering colors, and obscuring fine details, posing significant challenges for underwater imaging and rendering.

A.3 Backscatter Estimation

The backscatter estimation algorithm is outlined in algorithm 1, which calls algorithm 2 as a subroutine. A visual explanation of underwater attenuation is provided in Figure 5.

Algorithm 1: Estimate Backscatter

Input: $I_{sm}, z_{sm}, p_{dark} = 0.01, intervals\_num = 25, resized\_height = 300$

1. Resize I_{sm} to $resized\_height$
2. Set negative values in I_{sm} to 0
3. Resize z_{sm} to $resized\_height$
4. Set negative values in z_{sm} to 0
5. $darkZ, B_c \leftarrow$ getBackscatterByCurveFittingMultipleImages$(I_{sm}, z_{sm}, p_{dark})$ `(i.e., algorithm 2)`
6. $intervals \leftarrow$ linspace$(\min(darkZ), \max(darkZ), intervals_num)$
7. Initialize min_values_depth and $minvals$ as empty lists for each color channel
8. **for** each color channel k **do**
   (a) **for** each interval i in $intervals$ **do**
      i. Find indices ind within the current interval
      ii. **if** valid data exists **then**
         A. Append $\min(B_c[ind, k])$ to $minvals[k]$
         B. Append corresponding $darkZ$ value to $min\_values\_depth[k]$
9. Initialize out as a zero matrix
10. **for** each color channel k **do**
   (a) Fit model to $min\_values\_depth[k]$ and $minvals[k]$
   (b) Extract parameters b_∞ and b_{cb} into out
11. $B_\infty \leftarrow out[0, :]$
12. $B_\mathrm{b} \leftarrow out[1, :]$
13. **return** $\{B_\infty, B_\mathrm{b}\}$

Algorithm 2: Get Backscatter By Curve Fitting Multiple Images

Input: $I, z_{sm}, pdark = 0.01, rhoflag = 0, edges\_num = 10$

1. Initialize $edges$ with $edges\_num$ evenly spaced values between $\min(z_{sm})$ and $\max(z_{sm})$
2. Compute $zcluster =$ clusterRange$(z_{sm}, edges)$
3. Initialize $maskRho$ as a zero matrix with the same shape as z_{sm}
4. **for** i in range(len$(edges) - 1$) **do**
   (a) Set $thisMask = (zcluster == i)$
   (b) **if** $\sum(thisMask) > 0$ **then**
      i. Extract $thisRho$ using $thisMask$
      ii. Find darkest pixels in $thisRho$ with threshold $pdark$
      iii. Update $maskRho$ by adding $thisMaskRho$
5. Convert $maskRho$ to boolean
6. Extract darkest pixels and their corresponding depths using $maskRho$ and dm, resulting in $darkZ$ and Bc
7. **return** $darkZ, Bc$

A.4 Experimental Setup

We adopted hyperparameters consistent with those used in the original 3D Gaussian Splatting method, as they have shown to yield optimal results empirically. For underwater-specific parameters, we maintained the same learning rates as those used for color. Our data splits involved testing every 8 images, and we utilized the Adam optimizer.

- **Iterations:** 30,000
- **Position Learning Rate Initial:** 0.00016
- **Position Learning Rate Final:** 0.0000016
- **Position Learning Rate Delay Multiplier:** 0.01
- **Position Learning Rate Maximum Steps:** 30,000
- **Feature Learning Rate:** 0.0025
- **Direct Volume Absorption Learning Rate:** 0.0025
- **Backscatter Learning Rate:** 0.0025
- **Opacity Learning Rate:** 0.05
- **Scaling Learning Rate:** 0.005
- **Rotation Learning Rate:** 0.001
- **Lambda SSIM:** 0.3
- **Lambda Backscatter:** 0.1
- **Densification Interval:** 100
- **Opacity Reset Interval:** 3000
- **Densify From Iteration:** 1500
- **Densify Until Iteration:** 15,000
- **Densify Gradient Threshold:** 0.0002
- **Minimum Opacity Threshold:** 0.1