Deep Underwater Monocular Depth Estimation with Single-Beam Echosounder

Haowen Liu, Monika Roznere, and Alberto Quattrini Li

Abstract—Underwater depth estimation is essential for safe Autonomous Underwater Vehicles (AUVs) navigation. While there has been recent advances in out-of-water monocular depth estimation, it is difficult to apply these methods to the underwater domain due to the lack of well-established datasets with labelled ground truths. In this paper, we propose a novel method for self-supervised underwater monocular depth estimation by leveraging a low-cost single-beam echosounder (SBES). We also present a synthetic dataset for underwater depth estimation to facilitate visual learning research in the underwater domain, available at [https://github.com/bdacnw/sbes-depth](https://github.com/bdacnw/sbes-depth). We evaluated our method on the proposed dataset with results outperforming previous methods and tested our method in a dataset we collected with an affordable AUV. We further investigated the use of SBES as an additional component in our self-supervised method for up-to-scale depth estimation providing insights on next research directions.

I. INTRODUCTION

This paper presents an underwater depth estimation framework that is suited for Autonomous Underwater Vehicles (AUVs) with inexpensive sensor configurations, i.e., a monocular camera and a single-beam echosounder (SBES) – see Fig. 1.

Underwater depth estimation is an important task for AUVs in navigation, localization, and survey [1]. Typically AUVs use sensors, such as multi-beam sonar [2], photon beam [3], or LiDAR [4], but they are generally expensive, difficult to setup, and do not provide dense depth estimates, limiting their adoption. Cameras are an inexpensive solution that are available on many AUVs [1], [5]. However, camera-based depth estimation is challenging underwater compared to out-of-water; the underwater image formation is affected by light attenuation and backscattering due to suspended particulates and other environmental factors [5], [6].

Recent advances in deep learning have the potential to address such challenges [7]. Many of these deep monocular depth estimation methods have emerged for out-of-water domains [8]. They are generally trained on large well-established datasets such as KITTI [9] and NYU-depth [10]. However, there are no comparable datasets in the underwater domain to facilitate large-scale training for deep underwater depth estimation. While there are a few Generative Adversarial Network (GAN) based underwater depth estimation methods available (e.g., [11], [12]), they are still trained with out-of-water data augmented with synthetic underwater

The authors are with the Department of Computer Science, Dartmouth College, USA {haowen.liu.gr, monika.roznere.gr, alberto.quattrini.li} @dartmouth.edu

This work is in part supported by the Dartmouth Burke Research Initiation Award, NSF CNS-1919647, 2024541, 2144624.

©2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. DOI: 10.1109/ICRA48891.2023.10160459
frameworks. We will also discuss datasets required to train a deep learning framework well.

A. Underwater Depth Estimation

There has been an increasing number of studies on underwater depth estimation in recent years. Early works [13]–[15] used the underwater image formation model and the dark channel prior assumption to estimate the transmission map and backscattering constant for depth estimation. Li et al. [16] jointly estimated scene depth and restored the image from an underwater video sequence using stereo matching and fog information. Peng et al. [17] used both image blurriness and light absorption for estimating depth. For learning based methods, Gupta and Mitra [12] proposed a GAN based method for depth estimation of a single underwater image. Their method learns mapping functions between unpaired in-air RGB-D images and arbitrary underwater images to indirectly estimate depth images based on cycle-consistent learning. Similarly, Hambarde et al. [11] used GANs for coarse and fine-level depth map estimation, with synthetic underwater images constructed from in-air RGB-D images as input. Accurate up-to-scale depth estimation underwater is still an open problem.

B. Sensor-Camera Fusion

Monocular depth estimation using deep learning has primarily focused on in-land scenarios [18]–[21] than on underwater cases. Self-supervised monocular depth methods formulate depth estimation as an image reconstruction problem, where depth maps are used as an intermediate product that integrates into the image reconstruction loss [22]. It is well known that fusing data from other sensors can help increase the quality of depth estimates. Sensor configurations can be as simple as multiple cameras [23], [24], IMU [25]–[31], and sonar [32]. They can also be more complex and of higher cost, such as the setup by Richmond et al. [33] with multibeam sonar, fiber-optic gyroscope IMU, and doppler velocity log, or the SVIN2 [34] setup with profiling sonar, IMU, stereo camera, and pressure sensor. Within the learning-based literature, to improve and recover dense depth maps from monocular images, prior work have proposed to merge sparse depth measurements from other sensors, e.g., LiDAR. For example, Zhao et al. [35] used a multi-scale co-attention-guided graph propagation network adaptive to the sparsity patterns of sparse LiDAR depth input to better associate the spatial context with observed depth values. Park et al. [36] employed a novel architecture that additionally re-injects the LiDAR signal at the refinement stage to mitigate signal degradation due to normalization layers after concatenating image and LiDAR features at the start. Feng et al. [37] proposed a novel pseudo dense LiDAR representation to overcome the sparsity issue of a 4-beam LiDAR for more effective fusion with monocular image features. There has also been work on fusion of RGB images with single-row scanning automotive radars [38] or binaural echoes [39] for improving depth estimation. Our method on fusing echosounder with monocular images for depth prediction takes inspiration from these LiDAR based methods; however, one of the main differences is that, while the LiDAR provides multiple signals, the SBES provides a single value, thus requiring strategies to handle this challenge.

C. Underwater Dataset

There are very few existing datasets in the underwater domain, mainly targeted for SLAM or for image enhancement, compared to general out-of-water computer vision tasks. Examples include AQUALOC [40] which contains seabed recordings at different water depths from Remotely Operated Vehicles equipped with a monocular monochromatic camera, IMU, and a pressure sensor; the Underwater Caves Sonar [41] dataset consisting of data from a mechanically scanned imaging sonar, depth sensor, and camera imagery; and SQUID [42] containing images taken under varying water properties and respective 3D structures of the scenes. However, due to the challenges of underwater data collection, most underwater datasets only include a small number of samples obtained under specific settings and lack corresponding ground truth measurements, rendering them less suitable for deep learning based methods. While methods such as WaterGAN [7] proposed synthesizing underwater images via introducing artificial underwater effects to in-air images, they often introduce unwanted artifacts and fail to represent real underwater landscapes.

III. DATASET

Given the scarcity of underwater datasets and the lack of datasets comprising of echosounder measurements and monocular images, we created a dataset for underwater depth estimation models. Our dataset contains monocular and stereo images, semantic segmentations, and echosounder readings of realistic simulated underwater scenes to facilitate vision-based learning in the underwater domain. Note that stereo images were collected for future analysis, as here we focus on a monocular configuration. We describe the data creation and collection pipeline in the following.

To construct underwater scenes in simulation, we first selected models of various underwater structures, such as shipwrecks, reefs, and caves built using professional photogrammetry software from 3D model collection site SketchFab. As these models were constructed from real-world images, they provide accurate representation of structures found in real-world underwater environments. Then, each model is placed in a simulated underwater environment created using Unity game engine. The simulated environments are modified from an existing underwater simulator framework which contains custom shaders that incorporates a light transmission model, simulating underwater optical effects, thus providing a good amount of realism. As individual texture colors of the collected models vary greatly according to the color of their surrounding waters during the photogrammetry data collection process, color of the simulated water in each scene

https://sketchfab.com/
https://unity.com/
is sampled to match the texture color of the assigned model to look realistic. Table I summarizes the images collected from different categories of structures, and a sample of the different underwater scenes simulated can be found in Fig. 2.

![Figure 2: Examples of underwater scenes constructed in simulation](https://bluerobotics.com/store/rov/bluerov2/)

### A. Data Collection

We simulated an AUV, performing data collection in the underwater environments in Unity. The simulated AUV is equipped with an SBES and two RGB-D cameras in a stereo configuration with a baseline of 0.5 m. The echosounder has a range of 0.5 to 10 meters, and camera parameters such as resolution and focal length are kept the same as the actual camera used on a relatively inexpensive underwater robot, BlueRobotics BlueROV. Both the echosounder and cameras are placed at a pitch of 30-degree towards the sea floor to reduce the area of background water captured. For each scene, the AUV traveled to waypoints randomly generated within a bounding box around each scene. The path is calculated via three-dimensional A*. Each path was constrained to have length greater than a pre-defined distance and split in our experiments.

<table>
<thead>
<tr>
<th>Shipwreck(large)</th>
<th>Shipwreck(small)</th>
<th>Reef</th>
<th>Rock</th>
<th>Misc. Structure</th>
<th>Misc. wreck</th>
</tr>
</thead>
<tbody>
<tr>
<td>Train</td>
<td>10426</td>
<td>6019</td>
<td>2464</td>
<td>1366</td>
<td>2647</td>
</tr>
<tr>
<td>Val</td>
<td>586</td>
<td>390</td>
<td>180</td>
<td>129</td>
<td>210</td>
</tr>
<tr>
<td>Test</td>
<td>438</td>
<td>260</td>
<td>114</td>
<td>58</td>
<td>114</td>
</tr>
</tbody>
</table>

![Figure 3: Dataset instance with RGB image, depth map, and semantic label](https://bluerobotics.com/store/rov/bluerov2/)

**Table I**: Summary reporting the number of images of the proposed dataset and split in our experiments.

### IV. APPROACH

In this section, the proposed methodology is described in detail and the main components are outlined – an overview of the proposed approach is shown in Fig. 3.

#### A. Echosounder Fusion

While our method uses both monocular imagery and SBES measurements as input for dense depth estimation, it is difficult to fuse an echosounder measurement naively with image features as it is extremely sparse compared to the number of image pixels (1 vs >100k). Hence, we first attempt to address the sparsity issue via transforming the sparse measurement into a denser representation so that echosounder information can be more effectively fused with monocular image features.

To measure the distance to an object, an SBES emits an acoustic pulse - approximately in the shape of a cone and listens to the reflected pulses, using the time of flight for distance calculation. According to the echosounder and camera model described in our previous work [44], as shown in Fig. 4a, the base of the sound cone can be approximated as a circle with the center $cc_i$ in the camera reference frame as:

$$ cC_i = cC_E + m_i \cdot \tilde{v} $$

where $m_i$ is the echosounder distance reading, $cC_E$ is the echosounder’s location, and $\tilde{v}$ is the direction unit vector with respect to the camera reference frame. Given cone angle $a$, the radius of the circular region is calculated as:

$$ r_i = m_i \cdot \tan(a) $$

and with camera intrinsics $K$, assuming the pinhole camera model, we can identify the pixel coordinate $(u, v)$ that corresponds to the center of the same circular region, projected onto the image plane, as:

$$ \begin{bmatrix} u' \\ v' \end{bmatrix} = K \cdot cC_i $$

where

$$ u = \frac{u'}{w'} $$

$$ v = \frac{v'}{w'} $$

Although we now know that the echosounder measurement must correspond to the nearest point within the projected circle, it still covers a relatively large area on the image in our particular setting. Hence, we further reduced the candidate region by performing image segmentation to exclude background regions where the echosounder measurement is unlikely to pick up from. After preliminary experiments, we found a clustering algorithm based on Gaussian mixture model (GMM) [45] to perform well with underwater imagery. Finally, the remaining area was assigned the distance measurement returned by the echosounder while the rest are assigned 0, indicating an unknown value. This generated a mask of size $H \times W$, where $H$ and $W$ represent the height and width of the monocular image, respectively. The
C. Loss Functions

Self-supervised depth estimation can be expressed as a view-synthesis problem, where we try to predict a target image $I_t$ given the viewpoint of adjacent images $I_{t'}$. For successful synthesis of the target image from source images, we need to accurately estimate both depth $D_t$ and the relative pose between the target-source images. Hence, similar to [18], [19], photometric reprojection loss $L_p$ was used to estimate the pixel-level similarity between the target image $I_t$ and the synthesized target image $I_{t'\rightarrow t}$:

$$L_p = \sum_{t'} pe(I_t, I_{t'\rightarrow t})$$ (5)

and

$$I_{t'\rightarrow t} = I_t \langle \text{proj}(D_t, T_{t\rightarrow t'}, K) \rangle$$ (6)

where relative pose of source view $I_{t'}$ with respect to the target image $I_t$ is expressed as $T_{t\rightarrow t'}$, and $\text{proj}()$ is the projected coordinates of depth $D_t$ in $I_{t'}$ using camera intrinsics $K$. $\langle \rangle$ is the sampling operator. $pe$, the photometric reprojection error consisting of the L1 distance in pixel space and the SSIM loss [47], is calculated by:

$$pe(I_a, I_b) = \frac{\alpha}{2}(1 - \text{SSIM}(I_a, I_b) + (1 - \alpha)||I_a - I_b||_1)$$ (7)

where $\alpha$ is a hyper-parameter to weigh contributions of the two terms.

We also include the edge-aware smoothness $L_s$ [48] to encourage smooth depth estimations locally while preserving sharp boundaries:

$$L_s = |\partial_x \hat{d}_t| e^{-|\partial_x I_t|} + |\partial_y \hat{d}_t| e^{-|\partial_y I_t|}$$ (8)

where $\hat{d}_t = d_t / \bar{d}$ is the mean-normalized inverse depth.

Similarly to [18], we further applied a binary per-pixel mask $\mu$ to the photometric reprojection loss $L_p$ to filter out static frames and texture-less regions, which can be dominant in underwater scenes:

$$\mu = \min_{t'} pe(I_t, I_{t'\rightarrow t}) < \min_{t'} pe(I_t, I_{t'})$$ (9)

Finally, the overall loss $L$ is the linear combination of the photometric reprojection loss and smoothness loss:

$$L = \mu L_p + L_s$$ (10)

V. Experiments

We evaluated our method quantitatively and qualitatively on our proposed dataset as well as on real-world images. Results are presented in the following section.

A. Evaluation on Proposed Dataset

There are 25299 images and 1069 images used for training and validation, respectively. Our method was implemented in Pytorch and experiments were done on a workstation with a Nvidia GeForce RTX 3090 GPU. Images were scaled down to a size of $160 \times 224$, using batch size of 24 and an initial learning rate of $1e-4$ for training.

Table II shows the quantitative results on the proposed test set in comparison to existing methods, all trained on our


### Table II: Quantitative evaluation of our method against other monocular depth estimation methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>Superv.</th>
<th>Abs Rel</th>
<th>Sq Rel</th>
<th>RMSE</th>
<th>RMSE log</th>
<th>$\delta &lt; 1.25$</th>
<th>$\delta &lt; 1.25^2$</th>
<th>$\delta &lt; 1.25^3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>BTS [49]</td>
<td>S</td>
<td>0.2348</td>
<td>0.658</td>
<td>2.188</td>
<td>0.226</td>
<td>0.642</td>
<td>0.941</td>
<td>0.986</td>
</tr>
<tr>
<td>SfMLearner [19]</td>
<td>M</td>
<td>0.471</td>
<td>3.321</td>
<td>3.651</td>
<td>0.429</td>
<td>0.452</td>
<td>0.727</td>
<td>0.860</td>
</tr>
<tr>
<td>SC-Depth [50]</td>
<td>M</td>
<td>0.394</td>
<td>2.272</td>
<td>2.896</td>
<td>0.335</td>
<td>0.492</td>
<td>0.842</td>
<td>0.926</td>
</tr>
<tr>
<td>PackNet-Sfm [51]</td>
<td>M</td>
<td>0.319</td>
<td>1.302</td>
<td>2.437</td>
<td>0.291</td>
<td>0.574</td>
<td>0.871</td>
<td>0.945</td>
</tr>
<tr>
<td>Monodepth2 [18]</td>
<td>M</td>
<td>0.309</td>
<td>1.257</td>
<td>2.391</td>
<td>0.280</td>
<td><strong>0.587</strong></td>
<td>0.887</td>
<td>0.952</td>
</tr>
<tr>
<td>UW-Net [12]</td>
<td>GA</td>
<td>0.939</td>
<td>8.459</td>
<td>5.980</td>
<td>0.625</td>
<td>0.246</td>
<td>0.484</td>
<td>0.683</td>
</tr>
<tr>
<td>Ours</td>
<td>M</td>
<td><strong>0.309</strong></td>
<td><strong>1.230</strong></td>
<td><strong>2.356</strong></td>
<td><strong>0.277</strong></td>
<td>0.582</td>
<td><strong>0.897</strong></td>
<td><strong>0.957</strong></td>
</tr>
</tbody>
</table>

**Fig. 6:** A sample of qualitative comparison of Monodepth2 [18] and ours.

The existing methods include recent self-supervised and GAN-based methods. We included also a supervised method as a lower bound of performance, even if it would be difficult to obtain accurate ground truth dataset in the underwater domain. For all unsupervised methods that only predict depth up to an unknown scale factor, the resulting depths were scaled according to the ratio of echosounder distance measurement to the nearest depth within the projected echosounder sound cone to have a fair comparison. This strategy was used as it follows the measurement returned by a real echosounder used on our underwater robot, where the strongest signal return would determine the corresponding distance measurement, while dense ground truth depth used for median scaling cannot be obtained in this setting. From Table II as expected, we can see that BTS [49], with ground truth supervision, performed better, than other self-supervised method with ours being relatively close. Our method achieved positive results, outperforming (or ties) all other self-supervised and GAN-based methods, in almost all evaluation metrics. These results demonstrate that our self-supervised approach can still work reasonably well and is preferable than a supervised one, when a large dataset with accurate ground truth is not available.

Fig. 6 also shows some examples of depths maps produced for qualitative evaluation. Consistent with our quantitative results, our method produced the best depth maps overall. Compared to depth maps generated using Monodepth2 [18], our method had less incorrectly predicted background regions, see, e.g., top left corner of image in the fourth column. The echosounder mask encoding provided additional information that helped distinguish foreground and background regions, which were prone to error due to the lack of texture for photometric self-supervision. Also, compared to our method, Monodepth2 [18] tended to predict objects to be nearer than in reality, as seen in image 1 (second pillar from top left) and 5 (depression at near end of the boat). However, we notice that depth maps produced by our method were sometimes less sharp compared to [18], e.g., image 6, which may lead to loss of details. This may be due to the fact that the echosounder mask un-conservatively assigns the echosounder measurement to all regions within the mask, overlooking slight variances in depth.

### B. Supervision with Echosounder Measurement

In addition to indirectly providing information on ground truth measurements through echosounder mask encoding, we also explored the use of direct supervision using the echosounder measurement itself (Fig. 5b). Using the self-supervised losses described in the previous section, we were only able to estimate depth up to an unknown scale factor. Hence, to estimate metric depth, we included an additional L1 loss $L_b$ using the single echosounder distance measurement $h_t$ as the supervisory signal:
\[ L_b = \left| \frac{1}{k} \sum_{p \in P_k} D_t(p) - h_t \right| \] (11)

where \( P_k \) consists of the smallest \( k \) depths of regions within the echosounder mask. Average of the smallest \( k \) depths instead of the smallest depths was used to protect against outliers.

The total loss now becomes:

\[ L = L_p + L_s + L_b \] (12)

where each component is weighed using hyper-parameters accordingly.

Qualitative results of depth and absolute error maps produced using models trained with the new loss, compared to only using the echosounder mask are shown in Fig. 7. We can see that, in general, the quality of the depth maps produced by our method with echosounder loss is a little less sharp in comparison. However, in terms of absolute error maps, our method with echosounder loss performed better (2.258 vs 2.356). This suggests that by using the echosounder value as a supervisory signal does indeed allow our method to learn about the overall world scale to some extent, hence allowing it to outperform its counterpart without echosounder loss. Nevertheless, with only a single echosounder measurement with uncertain position, it proved to be difficult for our model to learn jointly the absolute scale and relative depth relationships between objects in the scene.

C. Evaluation on Real-World Images

Next, we carried out some qualitative evaluation on real-world data to demonstrate the transferability of our method trained using the proposed dataset. Real-world data collection were performed in the Caribbean Sea with a BlueROV2 installed with the Sony IMX273 camera and Ping echosounder. The camera has a resolution of 1.6 MP, a horizontal and vertical FOV of 96° and 72°. The echosounder has a beam width \( a \) of 30° and a maximum range set to 4 m. Actual images and their predicted depth maps are shown in Fig. 8. From the figure, we can see that despite the images being different from our dataset used for training, our model was still able to generalize relatively well to real-world images without fine-tuning. We also measured the inference time of a single image on a .8 GHz Intel i7 laptop with Nvidia Geforce 1660Ti, which is about 0.07 seconds per inference, suggesting feasibility for real-time application.

VI. CONCLUSION AND FUTURE STEPS

We presented the first deep underwater monocular depth estimation approach with single-beam echosounder input, as well as a new large-scale dataset for underwater vision tasks. We also validated our approach with evaluations in both the proposed dataset and real-world underwater images. Our proposed method provides potential in correcting the depth estimates with an inexpensive sensor configuration. The experiments provided some insights on next steps. We will investigate replacing the GMM algorithm for echosounder mask generation with a learnable network to capture closer objects in finer detail. We will also explore a denser integration of echosounder-camera data via including all the signal strengths and via a temporal network to improve the depth estimation which currently is limited due to the single supervisory signal from the echosounder. We also plan to expand our current dataset with more path variations, e.g., circular paths and dynamic objects such as fish, to enhance the realism of our dataset and cover scenarios that are encountered in the real world.


