2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Metrically Scaled Monocular Depth Estimation through Sparse Priors for Underwater Robots

Luca Ebner[1], Gideon Billings[2], Stefan Williams[2]

***Abstract*—In this work, we address the problem of real-time dense depth estimation from monocular images for mobile underwater vehicles. We formulate a deep learning model that fuses sparse depth measurements from triangulated features to improve the depth predictions and solve the problem of scale ambiguity. To allow prior inputs of arbitrary sparsity, we apply a dense parameterization method. Our model extends recent state-of-the-art approaches to monocular image based depth estimation, using an efficient encoder-decoder backbone and modern lightweight transformer optimization stage to encode global context. The network is trained in a supervised fashion on the forward-looking underwater dataset, FLSea. Evaluation results on this dataset demonstrate significant improvement in depth prediction accuracy by the fusion of the sparse feature priors. In addition, without any retraining, our method achieves similar depth prediction accuracy on a downward looking dataset we collected with a diver operated camera rig, conducting a survey of a coral reef. The method achieves real-time performance, running at 24 FPS on a NVIDIA Jetson Xavier NX, 160 FPS on a NVIDIA RTX 2080 GPU and 7 FPS on a single Intel i9-9900K CPU core, making it suitable for direct deployment on embedded GPU systems. The implementation of this work is made publicly available at https://github.com/ebnerluca/uw_depth.**

***Index Terms*—Marine Robotics, Computer Vision for Automation, Deep Learning for Visual Perception**

I. Introduction

Autonomous underwater vehicles often rely on visual based sensing for localization. For these systems, it is generally sufficient to build a sparse representation of the scene, using landmarks or detected feature points to optimize the vehicle pose estimates [1]. However, some tasks require a vehicle to operate near or interact with the environment, such as conducting close range imaging surveys or performing manipulation tasks. For such applications, a dense and real-time understanding of the scene is required for safe autonomous operation of the vehicle to enable obstacle avoidance and intervention planning. A practical solution to this problem must be robust to underwater imaging effects (Fig. 1), such as nonlinear attenuation, backscatter, and inconsistent lighting, must provide metrically accurate depth measurements, must be computationally efficient for real-time deployment on embedded compute systems, and ideally should be compact and low-cost for integration on diverse underwater robotic platforms.

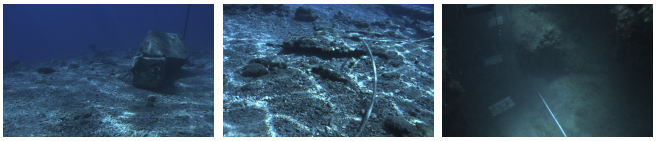

Fig. 1. Optical challenges in underwater imagery. Attenuation (left), reflections (middle) and turbidity (right).

Active-light depth sensors, such as LiDAR [2] or RGB-D cameras [3], [4], are commonly used in terrestrial applications and provide an intuitive solution for metrically accurate depth sensing. However, these sensors are challenging to deploy underwater, due to nonlinear, frequency dependent attenuation of light in the water column, refraction effects, and the scattering effects of suspended particulates. Imaging sonars provide an alternative solution that is robust to underwater lighting effects, and they are used in many marine field robotics applications [5]. However, they are expensive and provide limited spatial resolution compared to optical sensors with noisy depth estimates that make them less suitable for complex intervention and close range inspection tasks.

Passive stereo cameras provide a low-cost, off-the-shelf solution that is simple to setup and can provide dense, high-resolution depth measurements of the scene. However, stereo cameras can be bulky to integrate in underwater housings with an adequate camera baseline, making them difficult to deploy on small underwater systems or on the front of streamlined AUVs with highly limited space. Passive monocular cameras provide an attractive alternative to stereo cameras, due to their compactness and being readily integrated on any underwater platform. For this reason, predicting dense depth from monocular images for marine robotics has gained more attention in recent years, with significant advancements made in deep learning approaches.

In this work, we extend the recent advancements in learning based monocular depth estimation to demonstrate a practical approach for metrically accurate depth predictions that is suitable for deployment on diverse underwater platforms and is transferable to different camera systems without retraining, special calibration, or parameter tuning. In particular, we make the following contributions and demonstrations:

- An extension of a state-of-the-art lightweight network for dense depth prediction from monocular images to fuse triangulated feature points into the prediction stages

[1] Luca Ebner is with the Robotic Systems Lab, ETH Zurich and with the Australian Centre for Field Robotics, University of Sydney ebnerl@ethz.ch
[2] Gideon Billings and Stefan Williams are with the Australian Centre for Field Robotics, University of Sydney

through a dense parameterization of these sparse priors, improving the prediction accuracy and providing metric scale constraints.
- Training and evaluation of the method on the FLSea [6] dataset, demonstrating significantly improved performance through fusion of the sparse priors.
- Evaluation of the method on an additional image dataset from a coral reef survey we collected off of Lizard Island, Australia, with a downward facing diver operated camera rig. No images from this dataset or camera rig were used in the training of the method, demonstrating that the method generalizes well to different camera systems and environments.

II. Related Work

A. Monocular Underwater Depth Estimation

Some early approaches to underwater depth estimation from monocular images were driven by the need to enhance underwater images [7]. These methods aimed to mitigate the challenges posed by underwater lighting and attenuation effects, effectively transforming underwater images to resemble those taken in a terrestrial environment. To achieve this, researchers constructed models that explicitly account for the physics underlying underwater illumination and image formation. These methods, known as image formation models, incorporate coefficients to represent attenuation and other relevant water column properties. More recently, supervised deep learning approaches have shown great potential for learning image formation models and predicting depth maps from monocular images [8]. However, image formation models suffer from poorly constrained depth estimates and sensitivity to difficult imaging conditions such as turbidity and reflections.

Given the challenges and expense of collecting large-scale underwater datasets in diverse environments for training supervised models, another line of research has focused on generating realistic synthetic underwater datasets to train depth prediction models without the need for real images [9], [10]. While a promising avenue of research, the synthesis of underwater data is not yet able to fully capture the realistic formation of underwater images, leading to degraded performance on real data.

Recent work has achieved state-of-the-art performance through transfer learning, where a monocular depth estimation model is trained on large-scale terrestrial data and then fine tuned with a smaller underwater dataset. [11] provides a survey of these recent works, where popular adopted models include Monodepth2 [12], AdaBins [13], and U-Net [14] based architectures. Additionally, by incorporating domain specific loss formulations, the performance of the underlying models can be further improved for underwater environments [11]. Our method extends the work of AdaBins [13] and UDepth [11] in order to boost the prediction accuracy and provide metrically accurate scale through the fusion of sparse feature priors in the network architecture.

B. Sparse Measurement Fusion

An inherent problem of monocular camera systems is scale ambiguity. While supervised deep learning models are able to predict metric depth from monocular RGB image inputs without additional measurements, such models suffer from poor generalizability when the data distribution of the real world application does not match the distribution of the training data or the camera parameters change. Sparse depth priors from external measurements can be used to deliver a valuable depth prior and guidance signal for estimation in accurate metric scale, as surveyed in [15]. We provide a brief review here of learning based approaches for sparse prior fusion in depth prediction.

Categorized as *early fusion*, the first learning based methods simply concatenated the sparse depth maps and RGB image together as the network input [16], or convolved them in the first few layers of the model. In recent works, *late fusion* techniques have been developed with more complex architectures such as dual-encoder [17] or double encoder-decoder [18] models to better encode the sparse depth prior into the RGB image feature space. As a result, late fusion techniques generally outperform early fusion models at the cost of increased complexity. To cope with the high sparsity of the depth input, these methods often use either binary or continuous validity masks to guide the convolution operations to the relevant parts of the input prior. In the most recent approaches, researchers have begun to incorporate surface normal representations [19], affinity [20], and residual depth maps [21] into early, late or hybrid fusion architectures.

Many deep learning based methods for depth estimation from monocular images treat their problem in a one-shot fashion. At each inference step a new image is fed into the network to predict depth with no connection between subsequent frames or timesteps. In robotics however, these image inputs usually come from video streams, where there is a large correlation between sequential frames. Notably, many intervention robots are already capable of accurate localization through off-the-shelf visual-inertial methods like ORB-SLAM3 [22], which store knowledge about the environment in the form of sparse triangulated visual feature points. As proposed in methods like [23], such sparse depth priors can be used in fusion with corresponding RGB images to provide metric scale cues to a learning method, solving the problem of scale ambiguity and improving the inference accuracy. We draw inspiration from these prior works in our approach to fusing sparse feature based priors into the depth prediction network architecture, following a late fusion approach.

III. Method

A. Depth Prior Parameterization

To overcome the problem of scale ambiguity and to recover the global scale of the scene, we develop an approach to include sparse depth measurements of the scene in the form of visual features as input to the depth prediction network. We adapt the densified parameterization suggested by [23], which enables fusion of sparse priors with arbitrary density.

1) Sparse Prior Generation: By using the video stream from a robotic system, we can track and triangulate visual keypoints between frames. Popular visual-inertial SLAM methods like ORB-SLAM3 [22] track and triangulate features over many frames, resulting in highly accurate priors. As a simplification for our evaluation on the FLSea [6] dataset, we use the matched visual features between two consecutive frames and extract the depth value from the available dataset ground truth. These sparse measurements are then used as the prior guidance signal for the depth prediction task. The steps to generate a sparse set of prior depth measurements are as follows:

1) The image is divided into equal sized patches and SIFT keypoints [24] are detected in each patch.
2) The detected features are matched between frames by performing bidirectional nearest neighbor search followed by enforcing an epipolar constraint for outlier filtering. The epipolar constraint can be described as $k_i^1 = F^T k_i^2$, where (k_i^1, k_i^2) denotes the i-th keypoint match between frames $(1, 2)$ and F is the fundamental matrix.
3) For each inlier feature point, depth values are extracted from the ground truth depth maps at the keypoint location.

For evaluation on the LizardIsland dataset, we use the SLAM method developed in [25] to generate feature priors from the image sequence, where the feature positions and depths are projected from the constructed SLAM feature map, demonstrating the practical performance of the method as it would be deployed on an underwater vehicle with an online SLAM system.

2) Dense Prior Parameterization: Since the amount of available keypoints may vary from image to image, a generalized representation of constant size and suitable format is desirable to feed into the network. Inspired by [23], we define two maps $S_1(x, y)$ and $S_2(x, y)$ to map the sparse set of keypoints k_i into two continuous image representations.

Following the approach of [23], S_1 is a nearest neighbor interpolation of the sparse keypoint depths into a full size dense depth map, where every pixel (x, y) in $S_1(x, y)$ takes the depth value d_{k^*} of its closest keypoint k^*. The formula to compute S_1 is given as:

$$S_1(x, y) = d_{k^*}, \text{ where } k^* = \arg\min_i ||(x, y) - (x_i, y_i)||_2 \tag{1}$$

S_2 is formulated as a probability map to account for the distance of each pixel to its nearest keypoint. Let $r(x, y|k_i)$ be the euclidean pixel distance of point (x, y) to keypoint k_i. We assume the probability with regards to the pixel distance to a given keypoint to follow a normal distribution. The computation of S_2 then follows as:

$$S_2(x, y) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{r(x, y|k^*)^2}{2\sigma^2}\right), \tag{2}$$

where $r(x, y|k^*) = \min_i ||(x, y) - (x_i, y_i)||_2$ and σ is the standard deviation which is treated as a tuning parameter.

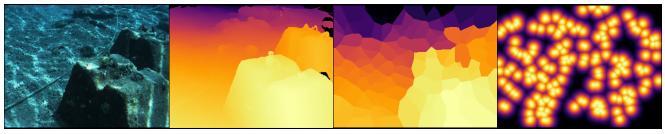

Fig. 2. Dense prior parameterization. RGB (first), ground truth depth data available from the given dataset (second), the S_1 nearest neighbor depth map (third) and the S_2 prior probability map (fourth), where the probability is highest (bright spots) at the location of the features.

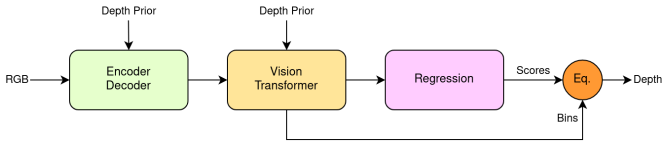

Fig. 3. Model overview with three main components based on AdaBins [13]: Encoder-decoder, vision transformer and regression. The depth prior is fed into the model at multiple stages during the decoder as well as in the beginning of the vision transformer.

As a result of empirical testing, we use $\sigma = 10$ for all our experiments.

To form the prior input to our model, the two maps S_1 and S_2 are depth-wise concatenated in the form of a two-channel image. This image is then fed into the network at multiple stages. Figure 2 shows an example of the parameterized prior inputs.

B. Architecture Design

The proposed network architecture is based on the popular AdaBins [13] model and the contributions of UDepth [11]. The architecture consists of a lightweight encoder-decoder backbone, an efficient vision transformer and a convolutional regression module. Figure 3 shows an overview of the model architecture.

1) Encoder Decoder Backbone: In the first module of the proposed network, we use a computationally fast encoder-decoder backbone. Motivated by the contributions of UDepth [11], we exchange the heavier encoder module of AdaBins [13] with a more efficient model based on MobileNetV2 [26]. This model is optimized for mobile architectures and lays the foundation for many applications which need real-time performance. Additionally, UDepth has demonstrated that this architecture attains good feature representations for underwater applications. The encoder-decoder module is constructed in a U-Net fashion [14]: In the encoder, we pass the input RGB image through all layers of the MobileNetV2 model and extract the signal at multiple stages for later use as skip connections. The decoder then consists of several upsampling layers which combine the current signal with skip connections from the encoder layers combined with the appropriately scaled depth parameterization.

2) Vision Transformer: While the preceding encoder-decoder block provides spatial feature information at multiple resolutions, the vision transformer component refines the results with global context. While vision transformers in general are computationally expensive [27], AdaBins introduced a lightweight vision transformer that we use with

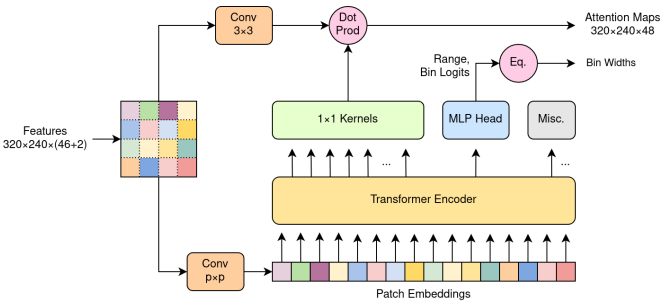

Fig. 4. Vision Transformer Structure based on AdaBins [13]. The feature image coming from the preceeding encoder-decoder module and depth parameterization are spatially divided into a number of patches of size $p \times p$. Using a convolutional block we have a set of patch embeddings as input sequence for the transformer encoder. After the encoder, one output embedding is used as input for a multilayer perceptron (MLP) head to predict the bin width logits and additionally range r, from which we then compute the metric bin widths via Equation 3. A longer subset of output embeddings is used as attention query kernels for forming the output attention maps via dot product.

some modifications [13]. Instead of estimating absolute depth values directly, they divide the depth range into n bins with adaptive size and predict the Softmax classification score for every pixel to belong to each of the bins. While we adopt this mechanism for estimating adaptive bin widths and range attention maps, our modified architecture additionally predicts the depth range r for each image. That is, we estimate the bin widths b_i by multiplying the normalized bin widths by the estimated range rather than a fixed upper bound, as was implemented in [13]. In our testing, we found that this modification to the network improves performance on inputs which are at different ranges than the training data.

To obtain the normalized bin widths $\bar{b}_i$, the adaptive bin logits $\tilde{b}_i$ output by the transformer are normalized such that they sum up to 1. For strictly positive bin widths, a small positive value $\epsilon = 10^{-3}$ is added to all logits. The formulation for computing the bin widths b_i is as follows:

$$b_i = r \cdot \bar{b}_i = r \cdot \frac{\tilde{b}_i + \epsilon}{\sum_j^n (\tilde{b}_j + \epsilon)} \text{ , where } i = 1, 2, ..., n \quad (3)$$

The bin width logits $\tilde{b}_i$ and range r are predicted through a multilayer perceptron (MLP) head with $n + 1$ outputs (n bins plus 1 for range r) from the first transformer output embedding. Figure 4 shows a visualization of our modified vision transformer module.

3) Convolutional Regression: The last stage of the network forms the depth image prediction from the bin centers and range attention maps output by the vision transformer module. The bin classification scores are computed following the approach of AdaBins [13], where the range attention maps from the vision transformer module are 1×1 convolved, followed by a Softmax activation to map the output logits into a set of n probabilities for each pixel, where n is the number of bins. The final depth image prediction $\hat{d}(x, y)$ is then formed as the linear combination of the probabilities $p_i(x, y)$ and bin centers c_i, as formulated in Equation 4:

$$\hat{d}(x, y) = \sum_i^n c_i \cdot p_i(x, y) \quad (4)$$

C. Loss Functions

The training loss is formulated as a combination of different loss functions, providing different constraints on the network predictions. In the following, we denote $\hat{d}$ as the predicted depth and d as the available ground truth.

1) Root Mean Squared Error: The traditional RMSE loss function directs the network to learn an exact metric scale prediction. This loss is formulated as

$$\mathcal{L}_{\text{RMSE}}(\hat{d}, d) = \sqrt{\frac{1}{N} \sum_i^N (\hat{d}_i - d_i)^2} \quad (5)$$

2) Scale Invariant Logarithmic Loss: For robotic intervention tasks errors are often calculated in logarithmic space [28], which penalizes errors at close range and becomes more forgiving at greater distances. Motivated by [11], [13], we utilize a parameterized variation of the Scale Invariant Logarithmic Loss (SILog Loss) [29], which allows us to balance the learning focus between estimating accurate metric scale and relative depth:

$$\mathcal{L}_{\text{SILog}}(\hat{d}, d) = \beta \sqrt{\frac{1}{N} \sum_i^N g_i^2 - \frac{\lambda}{N^2} (\sum_i^N g_i)^2} \quad (6)$$

where $g_i = \log \hat{d}_i - \log d_i$ and $\lambda \in [0, 1]$. As proposed by [13], we use $\lambda = 0.85$ and $\beta = 10$ in all of our experiments.

3) Chamfer Distance Loss: To guide the estimation of adaptive bin sizes b_i (described in Subsection III-B), we utilize a regularizer which encourages the distribution of bin centers to follow the actual depth distributions of a given input image. Following [13], we use the Chamfer Distance Loss first introduced by [30]:

$$\mathcal{L}_{\text{Chamfer}}(c, d) = \sum_{c_i \in c} \min_{d_j \in d} ||d_j - c_i||_2^2 + \sum_{d_j \in d} \min_{c_i \in c} ||d_j - c_i||_2^2 \text{ ,}$$
$$\text{where } i = 1, 2, ..., n, \; j = 1, 2, ..., N \quad (7)$$

4) Learning Objective: As our learning objective, we use a weighted combination of the losses $\mathcal{L}_{\text{RMSE}}$, $\mathcal{L}_{\text{SILog}}$ and $\mathcal{L}_{\text{Chamfer}}$, with $\lambda_1 = 0.3$, $\lambda_2 = 0.6$ and $\lambda_3 = 0.1$:

$$\mathcal{L}_{\text{Objective}} = \lambda_1 \cdot \mathcal{L}_{\text{RMSE}} + \lambda_2 \cdot \mathcal{L}_{\text{SILog}} + \lambda_3 \cdot \mathcal{L}_{\text{Chamfer}} \quad (8)$$

IV. Experiments

A. Datasets

1) FLSea: Our method is trained and evaluated on the FLSea dataset [6], which features forward looking imagery close to the sea floor in relatively shallow water depths less than 10 m. On average, the maximum distance range in the available ground truth is around 10 m. The publicly available

dataset consists of 22'451 image frames with RGB and metric ground truth depth data at 12 different underwater locations. Of these 12 locations, we use 10 for training and 2 for testing (*u_canyon* and *sub_pier*). FLSea showcases many of the lighting challenges described in Section II. Despite providing color corrected images, we use the raw images both during training as well as testing. Example pairs of RGB and depth map data from the FLSea dataset are shown in Figure 5.

As explained in Section I, obtaining true underwater depth is a very challenging task. For this, the ground truth depth maps of FLSea are generated using the Agisoft Metashape photogrammetry software [31]. Photogrammetry pipelines such as Metashape are considered state-of-art solutions for underwater 3D reconstruction in both research and industry. However, due to their extreme computational cost, such pipelines can not be deployed in online applications. Achieving offline photogrammetry accuracy with a real-time method is an ultimate aim of our research.

In a pre-processing step, we match features between frames and extract sparse prior points with their depth values for all images in the FLSea dataset. For this, we follow the procedure described in Subection III-A and store the keypoints as a list for later usage during training and testing.

2) LizardIsland Dataset: In order to test the proposed method's generalization to different camera systems and underwater environments not represented in the training data, we additionally evaluate the method performance on a dataset collected with a downward facing diver operated camera rig conducting a survey of a Coral Reef off of Lizard Island, Australia. The dataset is composed of a stereo image sequence which was processed through COLMAP [32] to obtain ground truth dense depth maps. The sequence was also processed through the SLAM method published in [25] to generate optimized feature maps that were projected back into the left camera image frame to obtain the sparse depth priors for input into the network. Example images from this dataset along with the sparse prior visualizations are shown with the evaluation results in Figure 6, where the left camera image was used as the monocular input to the network. The survey dataset includes 2'216 sequential images that were included in the evaluation of the method performance.

B. Learning Pipeline

For the implementation of this work we used the PyTorch [33] library. We initialize our model at random, except for the MobileNetV2 [26] part in our encoder-decoder module (see Subsection III-B). As our optimizer we choose AdamW [34] and train all 15.6 M network parameters freely in steps with a batch size of 6. For the learning rate $lr(t) = lr_0 \cdot r^t$ we use exponential decay over time, where $lr_0 = 0.0001$ is the base learning rate and $r = 0.9$ is the decay rate. The model is trained in a supervised fashion on training data from FLSea. For training robustness, we apply a series of random transforms to our inputs: Horizontal flips, color- and brightness scaling, as well as depth scaling. Throughout all training for these tests, we randomly select 200 depth priors per image frame for our dense depth parametrization (see Subsection III-A). The training loss is then computed between the prediction outputs and target depth maps by using the learning objective formulated in Subsection III-C. Using a NVIDIA RTX 2080 graphics card, the pipeline training took approximately 1 hour per epoch. The best results were achieved after around 10 hours of training.

C. Evaluation Metrics

To evaluate the performance of the trained models, we use a variety of standard error metrics. The error metrics between the predicted $\hat{d}$ and ground truth depth d are calculated as follows: For its straightforward interpretation, we choose standard RMSE in linear space (Eq. 9). Following the intuition in Subsection III-C, we further evaluate on RMSE in logarithmic space (Eq. 10) to investigate emphasized errors at close range. We also compare this error against its scale invariant counterpart (Eq. 11) to analyze the accuracy of the overall estimated scene scale.

$$\text{RMSE}_{\text{lin}}(\hat{d}, d) = \sqrt{\frac{1}{N}\sum_{i}^{N}(\hat{d}_i - d_i)^2} \tag{9}$$

$$\text{RMSE}_{\log}(\hat{d}, d) = \sqrt{\frac{1}{N}\sum_{i}^{N}(\log \hat{d}_i - \log d_i)^2} \tag{10}$$

$$\text{RMSE}_{\text{silog}}(\hat{d}, d) = \sqrt{\frac{1}{N}\sum_{i}^{N}(\log \hat{d}_i - \log d_i + \alpha(\hat{d}, d))^2}, \tag{11}$$

where $\alpha(\hat{d}, d) = \frac{1}{N}\sum_{i}^{N}(\log d_i - \log \hat{d}_i)$ is the term that makes the error scale invariant [29].

Furthermore, we analyze the Mean Absolute Relative Error to gain insight into the relative accuracy of our prediction:

$$\text{MARE}(\hat{d}, d) = \frac{1}{N}\sum_{i}^{N}\frac{|\hat{d}_i - d_i|}{|d_i|} \tag{12}$$

D. Results

1) FLSea: We evaluate the performance of the network on the test split of the FLSea [6] dataset and analyze how the prediction improves by using sparse priors. As when training, we use 200 randomly selected sparse priors per image which are stored from our pre-processing step. Figure 5 shows a visualization of our predictions and Table I presents the evaluated error metrics, cut off at different ranges to provide insight into near depth and far depth prediction accuracy.

The results of Table I demonstrate significantly improved depth prediction accuracy for all depth ranges and across all evaluation metrics through the fusion of sparse feature priors. It is important to reiterate that the ground truth depth maps of the FLSea [6] dataset were computed through a photogrammetric 3D reconstruction pipeline, so these metrics represent the ability of our method to reproduce these photogrammetry results. However, as discussed in Subsection IV-A, such pipelines are recognised as state-of-art solutions for both research and industry.

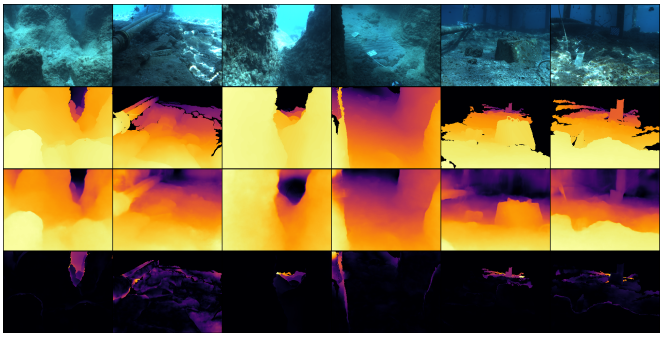

Fig. 5. Depth prediction examples on the FLSea dataset. RGB input (first row), ground truth depth map (second row), predicted depth map (third row) and absolute error (fourth row). Despite various optical challenges such as attenuation, reflections and turbidity our approach yields stable predictions.

Table I. Evaluation metrics of our model tested on the test split of the FLSea dataset. The results show a comparison between UDepth [11], a rescaled version of UDepth, and our method. The rescaled version of UDepth represents a simple way to recover the scene scale s by minimizing the least squares error at the keypoint locations such that $s = \arg\min_s \sum_i (\hat{d}_i * s - d_i)^2$, where i are the keypoint locations. When examining the results, it is evident that the fusion of depth priors in early stages of our method is superior to mere rescaling by an image-wise constant factor s.

Range	Model	RMSE (Linear)	RMSE (Log)	RMSE (SILog)	MARE
Full range	UDepth	0.660	0.181	0.146	0.137
	UDepth scaled	0.580	0.145	0.140	0.103
	Ours	0.372	0.089	0.087	0.037
$d < 5\,\mathrm{m}$	UDepth	0.385	0.158	0.116	0.128
	UDepth scaled	0.318	0.125	0.110	0.098
	Ours	0.167	0.057	0.055	0.031
$d < 1\,\mathrm{m}$	UDepth	0.266	0.245	0.073	0.264
	UDepth scaled	0.204	0.199	0.082	0.191
	Ours	0.040	0.042	0.032	0.029

2) LizardIsland: We evaluated the performance of the method on the LizardIsland dataset, with the model trained only on the FLSea dataset. Figure 6 shows visualizations of the predictions for a few samples from this dataset, and Table II gives the quantitative evaluation of the depth prediction accuracy. In addition to evaluating the predicted depth map accuracy, we evaluate the accuracy of the nearest neighbor prior depth maps with the same evaluation metrics. The network predictions show significant improvement over the naive nearest neighbor depth maps and achieve similar accuracy on the LizardIsland sequence as they do on the test sequences of the FLSea dataset, showing that the method can effectively generalize to novel underwater environments and camera systems not included in the training data.

Table II. Evaluation of the proposed method on the LizardIsland dataset, with the model trained only on the FLSea [6] dataset. The evaluated depth range was limited to $5\,\mathrm{m}$. Mean accuracy of the predicted depth maps (top row) versus standalone input nearest neighbor prior (bottom row).

	RMSE (Linear)	RMSE (Log)	RMSE (SILog)	MARE
Prediction	0.0632	0.0439	0.0417	0.0288
Prior	0.0715	0.0519	0.0500	0.0304

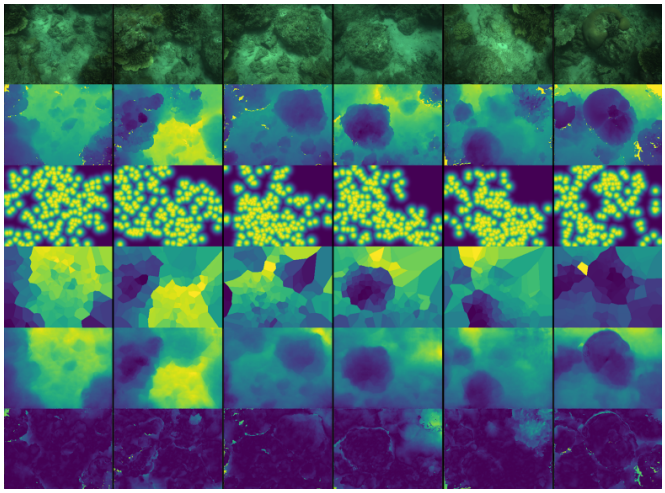

Fig. 6. Visualization of prediction results on samples from the LizardIsland dataset. RGB input (first row), ground truth depth map (second row), prior probability distribution map (third row), prior nearest neighbor map (fourth row), predicted depth map (fifth row) and absolute error between the prediction and ground truth depth maps (sixth row).

V. Conclusion

In this work, we have demonstrated a compact and cost-efficient dense depth sensing solution for mobile underwater robots. Our method follows a supervised monocular deep learning approach, where we fuse sparse depth measurements in order to solve the problem of scale ambiguity and improve the depth prediction accuracy. Experiments conducted on challenging underwater datasets demonstrate significantly improved depth prediction performance through fusion of the sparse priors. With a RMSE of $0.167\,\mathrm{m}$ ($5\,\mathrm{m}$ range) and $0.040\,\mathrm{m}$ ($1\,\mathrm{m}$ range) on the FLSea [6] dataset, we achieve promising accuracies for using this method in underwater intervention and close range survey applications. However, some intervention tasks may require higher levels of precision, motivating further research to improve this method. Considering that our approach requires only a monocular camera and some knowledge about the robot's motion, our solution is easy to integrate and particularly attractive for low-cost and small sized vehicles. Regarding computational efficiency, our proposed model remains lightweight and achieves real-time framerates at $24\,\mathrm{FPS}$ on a NVIDIA Jetson Xavier NX GPU, $160\,\mathrm{FPS}$ on a NVIDIA RTX 2080 gpu and $7\,\mathrm{FPS}$ on an Intel i9-9900K cpu using a single core.

In an outlook for future work, we will explore replacing the naive nearest neighbor approach of the depth prior parametrization with a smoother representation or better informed guess, such as propagating the previous prediction as an input for the current prior guess. In a similar matter, we would seek to integrate knowledge about the sparse depth priors' uncertainty which is often available when using SLAM pipelines. Furthermore, we would like to incorporate a segmentation for points at infinity as a side task to handle such areas more robustly.

REFERENCES

[1] J. A. Placed, J. Strader, H. Carrillo, N. Atanasov, V. Indelman, L. Carlone, and J. A. Castellanos, "A survey on active simultaneous localization and mapping: State of the art and new frontiers," *IEEE Transactions on Robotics*, 2023.

[2] D. McLeod, J. Jacobson, M. Hardy, and C. Embry, "Autonomous inspection using an underwater 3d lidar," in *2013 OCEANS-San Diego*. IEEE, 2013, pp. 1–8.

[3] H. Lu, Y. Zhang, Y. Li, Q. Zhou, R. Tadoh, T. Uemura, H. Kim, and S. Serikawa, "Depth map reconstruction for underwater kinect camera using inpainting and local image mode filtering," *IEEE Access*, vol. 5, pp. 7115–7122, 2017.

[4] S. T. Digumarti, G. Chaurasia, A. Taneja, R. Siegwart, A. Thomas, and P. Beardsley, "Underwater 3d capture using a low-cost commercial depth camera," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1–9.

[5] K. Sun, W. Cui, and C. Chen, "Review of underwater sensing technologies and applications," *Sensors*, vol. 21, no. 23, p. 7849, 2021.

[6] Y. Randall and T. Treibitz, "Flsea: Underwater visual-inertial and stereo-vision forward-looking datasets," 2023.

[7] J. Zhou, T. Yang, and W. Zhang, "Underwater vision enhancement technologies: A comprehensive review, challenges, and recent trends," *Applied Intelligence*, vol. 53, no. 3, pp. 3594–3621, 2023.

[8] P. Hambarde, S. Murala, and A. Dhall, "Uw-gan: Single-image depth estimation and image enhancement for underwater images," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–12, 2021.

[9] Q. Zhao, Z. Zheng, H. Zeng, Z. Yu, H. Zheng, and B. Zheng, "The synthesis of unpaired underwater images for monocular underwater depth prediction," *Frontiers in Marine Science*, vol. 8, p. 690962, 2021.

[10] P. G. O. Zwilgmeyer, M. Yip, A. L. Teigen, R. Mester, and A. Stahl, "The varos synthetic underwater data set: Towards realistic multi-sensor underwater data with ground truth," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 3722–3730.

[11] B. Yu, J. Wu, and M. J. Islam, "Udepth: Fast monocular depth estimation for visually-guided underwater robots," 2023.

[12] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3828–3838.

[13] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4009–4018.

[14] N. Siddique, S. Paheding, C. P. Elkin, and V. Devabhaktuni, "U-net and its variants for medical image segmentation: A review of theory and applications," *Ieee Access*, vol. 9, pp. 82 031–82 057, 2021.

[15] J. Hu, C. Bao, M. Ozay, C. Fan, Q. Gao, H. Liu, and T. L. Lam, "Deep depth completion from extremely sparse data: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[16] F. Ma and S. Karaman, "Sparse-to-dense: Depth prediction from sparse depth samples and a single image," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 4796–4803.

[17] M. Jaritz, R. De Charette, E. Wirbel, X. Perrotton, and F. Nashashibi, "Sparse and dense data with cnns: Depth completion and semantic segmentation," in *2018 International Conference on 3D Vision (3DV)*. IEEE, 2018, pp. 52–60.

[18] J. Tang, F.-P. Tian, W. Feng, J. Li, and P. Tan, "Learning guided convolutional network for depth completion," *IEEE Transactions on Image Processing*, vol. 30, pp. 1116–1129, 2020.

[19] J. Qiu, Z. Cui, Y. Zhang, X. Zhang, S. Liu, B. Zeng, and M. Pollefeys, "Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3313–3322.

[20] X. Cheng, P. Wang, and R. Yang, "Depth estimation via affinity learned with convolutional spatial propagation network," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 103–119.

[21] J. Gu, Z. Xiang, Y. Ye, and L. Wang, "Denselidar: A real-time pseudo dense depth guided depth completion network," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1808–1815, 2021.

[22] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[23] Z. Chen, V. Badrinarayanan, G. Drozdov, and A. Rabinovich, "Estimating depth from rgb and sparse sensing," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 167–182.

[24] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[25] G. Billings, R. Camilli, and M. Johnson-Roberson, "Hybrid visual slam for underwater vehicle manipulator systems," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6798–6805, 2022.

[26] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[27] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, "A survey on vision transformer," *IEEE transactions on pattern analysis and machine intelligence*, vol. 45, no. 1, pp. 87–110, 2022.

[28] A. Mertan, D. J. Duff, and G. Unal, "Single image depth estimation: An overview," *Digital Signal Processing*, vol. 123, p. 103441, 2022.

[29] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[30] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 605–613.

[31] Agisoft, "Metashape photoscan professional," 2018. [Online]. Available: https://www.agisoft.com/downloads/installer/

[32] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[34] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=Bkg6RiCqY7