Self-Supervised Underwater Stereo Depth Estimation

Mukul Chodhary[1]

***Abstract*— Acquiring high-quality training data for underwater depth estimation has posed numerous challenges, and due to the domain gap, in-air models cannot be directly applied to underwater scenarios. To tackle this problem, we introduce a pipeline that enables a novel transfer learning-based method to fine-tune stereo depth estimation models specifically for underwater environments. In-air methods have already shown significant improvement by incorporating synthetic labels for unlabelled data produced using existing depth estimation models. We incorporate this approach in our pipeline to generate stereo pairs from monocular datasets, utilizing a fine-tuned underwater monocular model to produce ground truth labels.**

To evaluate the zero-shot capabilities of our approach, we randomly selected 100 images from each of the four publicly available datasets provided in FLSea [1]. Our fine-tuned stereo model demonstrates improved zero-shot performance on these datasets, showcasing increased depth resolution and accuracy in underwater scenarios.

I. Introduction

Underwater imaging presents challenges due to water's unique optical properties, including light absorption, scattering, and refraction. These properties degrade image quality, characterized by low contrast, color distortion, and blurring, complicating accurate depth estimation. In both terrestrial and underwater environments, depth estimation plays a pivotal role in various applications, ranging from autonomous navigation to underwater exploration and environmental monitoring.

Traditionally, depth estimation techniques rely on monocular or stereo-vision approaches. Monocular depth estimation (MDE), inferring depth from a single image, is favored for its simplicity and low computational cost. However, underwater, MDE faces significant challenges. The absence of stereo cues and optical distortions degrade depth accuracy, leading to unreliable results.

Stereo depth estimation (SDE), leveraging two or more images from different viewpoints, offers a promising approach for underwater depth estimation. By exploiting binocular disparities, stereo methods can mitigate some limitations of MDE. However, stereo techniques also face underwater challenges. Water's refractive properties introduce distortions, complicating image matching and depth estimation accuracy.

Collecting high-quality training data for monocular and stereo depth estimation in underwater environments is challenging. Unlike terrestrial settings with readily available datasets, underwater data collection is complex and costly, requiring specialized equipment and often human divers, adding logistical challenges and expenses.

Underwater environments present further difficulties, such as poor visibility, dynamic conditions, and interference from marine life. Traditional methods like sonar struggle with low resolution due to slow scanning rates [3], and LiDAR faces signal attenuation and scattering [4]. Structure from Motion (SfM) offers potential solutions by using overlapping images to reconstruct 3D structures, but its underwater application is complicated by water clarity and lighting conditions. SfM ground truth data often lacks detail, making precise depth estimation challenging, necessitating robust methods to ensure reliable depth estimation despite environmental obstacles.

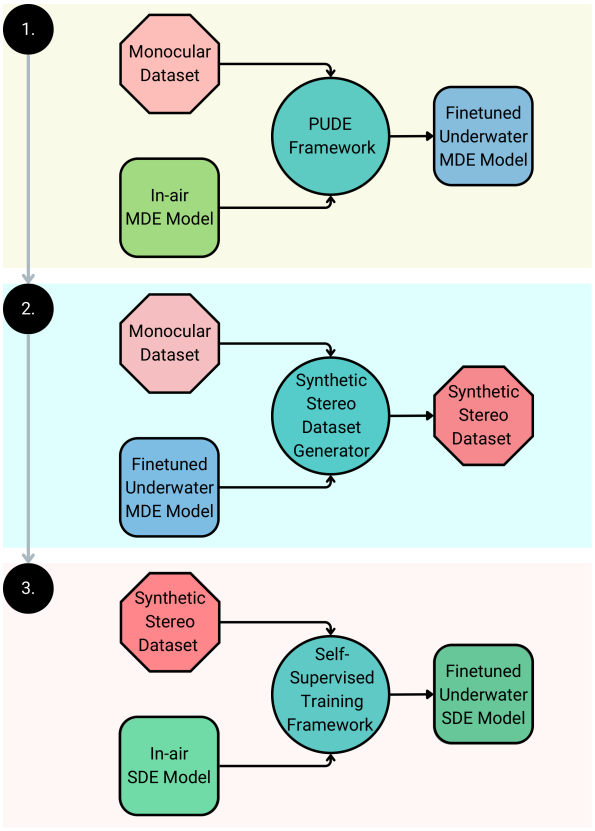

Fig. 1: High-level pipeline for fine-tuned underwater stereo depth estimation model.

Depth estimation techniques tailored to underwater environments are needed. These techniques must consider water's unique optical characteristics, address limitations of existing monocular and stereo approaches, and overcome data collection challenges. Recent in-air depth estimation advancements [2,5] have shown significant improvements using synthetic ground truth data from state-of-the-art models, and similar techniques can improve underwater depth estimation.

We present a pipeline, illustrated in Fig. 1, utilizing in-air monocular depth estimation models to enhance underwater stereo depth estimation performance.

[1]1172562, Department of Electrical and Electronic Engineering, University of Melbourne, Australia

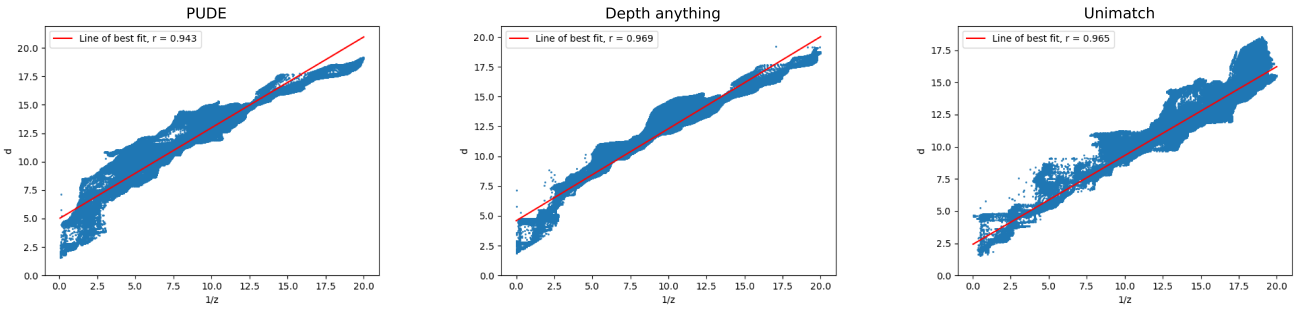

Fig. 2: Linearity comparison between monocular models PUDE, Depth anything [2] and stereo model Unimatch on a random image.

We contribute:

- A fully automatic pipeline to produce a fine-tuned underwater stereo depth estimation model using in-air monocular and stereo models.
- A robust point selection algorithm for parameter estimation in the PUDE framework [6].
- Combining the PUDE training framework with the "learning stereo from single images" training framework [5] to create a self-supervised training framework for the stereo model.

By integrating these components, our approach addresses underwater depth estimation challenges, leveraging advancements in in-air techniques to improve depth accuracy and reliability in underwater environments.

II. Background

A. Underwater image formation model

As light travels through water, it interacts with suspended particles, dissolved substances, and water molecules, resulting in scattering and absorption. Shorter wavelengths, like blue and green light, scatter more strongly than longer wavelengths, giving underwater scenes their blue or green hue. Light attenuation varies with depth and water quality. Clear oceanic waters experience absorption mainly by water molecules and dissolved substances, while turbid or coastal waters show significant scattering and absorption by suspended particles. Accurate underwater image formation modeling requires considering these optical properties and their spatial and spectral variations. A common underwater image formation model uses background light to approximate the true in-scattering term [7].

$$I_{c,i} = J_{c,i} \cdot t_{c,i} + B_c^{\infty}(1 - t_{c,i}) \tag{1}$$

and

$$t_{c,i} = e^{-\beta_c \mathbf{z}_i} \tag{2}$$

Here $I_{c,i}$ and $J_{c,i} \in [0, 1]$ are the underwater and in-air colour values at a pixel i. The $J_c, I_c \in \mathbb{R}^{HW}$, subscript c are the index of color channels $\{R, G, B\}$. The medium transmission rate $t_{c,i}$, is associated with the beam attenuation coefficient β_c and the depth $\mathbf{z}_i$. B_c^{∞} is the ambient light backscattering coefficient.

B. In-air depth estimation models applied to underwater

$$\mathbf{z}_i = \frac{1}{s\mathbf{d}_i + h}, \forall i \in \{1, \ldots, HW\}. \tag{3}$$

The $\mathbf{z}_i$ is the metric depth, and $\mathbf{d}_i$ is the inverse relative depth at pixel i. The parameters s and h are two scales and shift coefficients. We observe that the current monocular models, DPT [8], and PUDE, have a good linear profile for d vs $1/z$ but struggle in the extremes, i.e. for close and too far regions. The Stereo models have a better linear profile than monocular but these suffer from high variance, Fig. 2.

C. PUDE

The PUDE training framework consists of two main components: physics-informed knowledge transfer and direct knowledge transfer. PUDE initializes teacher and student models with the same in-air model, DPT. In this framework, the teacher model's depth estimations are treated as ground truth. These estimates are used to determine the underwater parameters in Eq. 1 and 2, then used to evaluate image formation model-based bound losses.

These bound losses transfer the in-air depth estimation model, DPT, to underwater settings. The loss functions improve in-air estimation in both near and background regions by considering the bounds of the underwater image formation models. For completeness, we describe these bound losses in Sec. III-E.3.

D. Unimatch

Unimatch [9] is a versatile model that enables cross-task transfer between optical flow, rectified stereo matching, and unrectified stereo depth estimation. It outperforms or compares favourably to recent state-of-the-art methods across 10 popular flow, stereo, and depth datasets, while also being simpler and more efficient in both model design and inference speed. Consequently, we selected Unimatch as the student stereo model for the third step of our pipeline. This selection allows us to fine-tune a single in-air model to achieve benefits across all three tasks. However, in this work, we focus solely on evaluating the results of stereo-matching or inverse depth prediction.

E. Image Inpainting

Inpainting methods aim to reconstruct missing or corrupted parts of an image based on the surrounding information. These techniques range from simple methods like linear interpolation to more sophisticated algorithms such as exemplar-based or patch-based inpainting [10–13]. In recent years, deep learning-based inpainting approaches have gained popularity due to their ability to learn complex image patterns and structures. These deep learning models, such as Transformers, Generative Adversarial Networks (GANs) or convolutional neural networks (CNNs), have outperformed many traditional techniques regarding inpainting quality and efficiency [14–19].

We opted for the Navier-Stokes-based inpainting method implemented in OpenCV [20], a classical inpainting technique. According to [5], inpainting quality does not significantly impact stereo-matching outcomes. Hence, we chose this faster and more general classical inpainting method.

III. Method

A. Overall Pipeline

The overall pipeline is split into three parts as outlined in Fig. 1:

1. We utilise the PUDE training framework to fine-tune the Depth Anything model, which has shown greater generalisability than DPT. By integrating Depth Anything into the PUDE framework, we leverage its robust performance and adaptability, further improving underwater depth estimation. We also enhance the dark point selection algorithm for greater robustness, using these points to evaluate underwater parameters (Sec. III-C).
2. We create a synthetic stereo dataset (Sec. III-D) using the fine-tuned MDE model and a real monocular dataset, treating the fine-tuned underwater MDE model's estimation as the ground truth.
3. We employ a self-supervised framework (Sec. III-B) to fine-tune the state-of-the-art SDE model, Unimatch [9], using the synthetic data for underwater scenarios.

B. Self-supervised training framework overview

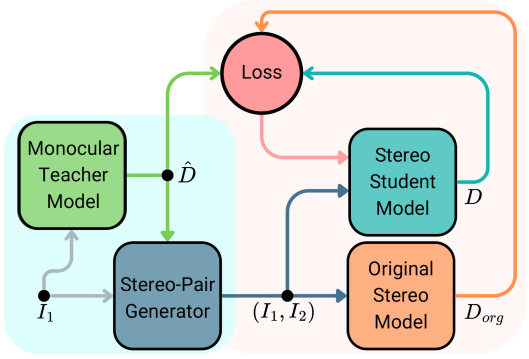

Fig. 3: Self-supervised Training Framework

The training framework aims to leverage the unique advantages of both monocular and stereo depth estimation models. Monocular models often exhibit higher accuracy and benefit from a larger dataset than stereo models. Conversely, stereo models are more general and possess excellent linearity properties.

In our framework, we use the estimations from the underwater MDE models as ground truth to synthesize stereo image pairs from monocular datasets. This approach eliminates the need for a stereo dataset for new environments, allowing us to utilize a variety of underwater monocular datasets effectively. We draw insights from [5], incorporating a stereo-student model and a stereo-pair synthesis layer to address issues related to occlusions and collisions. [5] also demonstrated that increasing the amount of monocular-derived training data enhances stereo performance and accuracy.

C. Underwater parameter estimation

1) Point selection algorithm: To estimate the parameters in the underwater image formation model (Eq. 1), we select the darkest points evenly across the image as outlined in Alg. 1. This set of selected points is denoted as M To do so, we iterate through the darkest pixels and select at most $cluster\_size$ pixels within $cluster\_radius$ of the selected point until we have selected N points. This approach forces even selection of unique points around the image. Only green and blue channels are used as UDCP [21] suggested they are sufficient in underwater cases due to their resilience against the absorption and scattering of light in water.

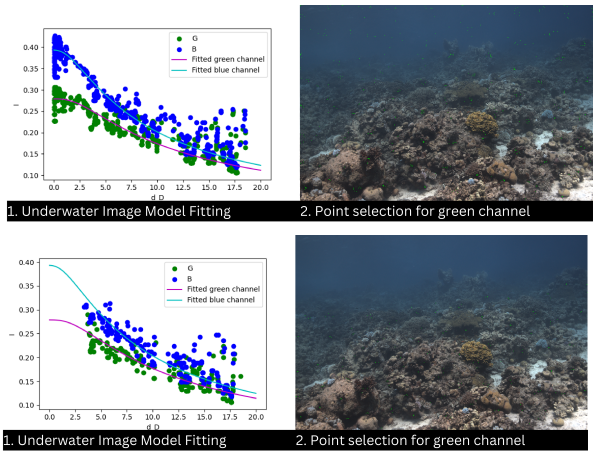

Fig. 4: Point selection and model fitting for two iterations. In the first iteration (first row) the points are selected evenly across the image, and in the second iteration (second row) points are selected from nearby regions after transmission rate thresholding.

2) Minimisation problem formulation: The physics-based perimeter estimation of the underwater image formation model is very sensitive to the selected points, hence it is necessary to ensure the previous step is done well.

As outlined in the previous step, we are interested in extracting the darkest points evenly for parameter extraction. This is mainly to simplify the underwater image formation

Algorithm 1 find_set_M Algorithm

Input: $I, d, N, cluster\_radius, cluster\_size, M\_range$
Output: M
for G, B channel **in** I **do**
    Sort pixel values for $channel$ to obtain darkest indices
    Determine percentiles of depth values within M_range
    while number of selected pixels $< N$ **do**
        for pixel **in** remaining darkest indices **do**
            Count number of selected pixels which are within $cluster\_radius$ of the current pixel
            Check if depth value is within percentile range and counted pixels don't exceed cluster radius
            if conditions are satisfied **then**
                Add current pixel to $selected\_pixels$
            end if
            if N pixels are selected **then**
                Remove outlier points
                if N pixels are selected **then**
                    Break
                end if
            end if
        end for
        if N pixels are still not selected **then**
            Adjust parameters (*M_range, cluster_radius, cluster_size*) to refine selection criteria
        end if
        M.append($selected\_pixels$)
    end while
end for
Return M array for all channels

model, i.e. $J_{c,i} \approx 0$ which means for all

$$I_{c,i} \approx B_c^{\infty}(1 - t_{c,i}) \tag{4}$$

To simplify the least-square problem, we modify the medium transmission rate by introducing two temporary parameters $\nu_c = \beta_c/s$ and $\mu = h/s$. With the teacher model's estimation d_i^P,

$$t_{c,i} = e^{-\frac{\nu_c}{\mathbf{d}_i^T + \mu}} \tag{5}$$

Denote $M_c, c \in \{G, B\}$, be the set of selected points from the point selection algorithm Alg.1. We define $\epsilon_c(M_c)$, the error for set M_c by substituting Eq.5 into Eq.4.

$$\epsilon_c(M_c) := \sum_{i \in M_c} \|I_{c,i} - B_c^{\infty}(1 - e^{-\frac{\nu_c}{\mathbf{d}_i^T + \mu}})\|_2^2 \tag{6}$$

We then solve the following least square problem by summing over green and blue channels to obtain the estimate of the parameters:

$$(\hat{\nu_G}, \hat{\nu_B}, \hat{\mu}, \hat{B_c^{\infty}}) = \underset{\nu_G, \nu_B, \mu, B_c^{\infty}}{\operatorname{argmin}} \sum_{c \in \{G,B\}} \epsilon_c(M_c) \tag{7}$$

3) Parameter estimation algorithm: The parameter estimation algorithm, Alg. 2, performs the minimisation problem twice in Eq.7. In the second optimisation run, we freeze the backscattering coefficient, $\hat{B}_c^{\infty}$ to refine the estimates of other parameters. The transmission rate threshold t_thresh, allows us to remove the regions where their prediction is unreliable. We remove all pixels for a channel where $\tilde{t}_{c,i} < 0.1$, these correspond to pixels in the background regions which are furthest from the observer.

The other parameters were chosen based on observations during experiments to achieve the best parameter estimation results, $N = [500, 200]$, $freeze\_B = [False, True]$, $t\_thresh = 0.1$, $cluster\_size = [5, 2]$, $M\_range = [5\%, 100\%]$, $cluster\_radius = [48, 40]$. The parameters for Alg. 1 were chosen to maximise the even selection of the darkest points in the target image.

Algorithm 2 Parameter Estimation Algorithm

Input: I, d^T
Hyper-params: N, $freeze\_B$, $cluster\_radius$, $cluster\_size$, M_range, $scale\_factor$, $inital\_values, t\_thresh$
Output: $\hat{\nu}_G, \hat{\nu}_B, \hat{\mu}, \hat{B}_c^{\infty}$
    /*Initialise params*/
$params \leftarrow initials\_values$
$I' \leftarrow$ I
    /*Find params iteratively*/
for i **in** range(len(N)) **do**
    $M\_i \leftarrow$ FIND_SET_M(I', , d^T, $N[I]$, $cluster\_size[i]$, $cluster\_raidus[i]$, M_range)
    $params \leftarrow$ SOLVE_MINIMISATION_PROBLEM(I',d^T, M_i, $params$, $freeze\_B[i]$)
    $I' \leftarrow$ THRESHOLD_ON_T($I, d^T, params, t\_thresh$)
end for
return $params$

D. Stereo-pair generation

Utilising monocular depth networks for generating training data for stereo networks without relying on stereo pairs or ground truth disparity has many advantages. This approach enhances data efficiency by eliminating the need for paired stereo images, reducing data collection efforts and costs, which is crucial in underwater depth estimation due to the difficulty of collecting underwater images.

It also offers scalability, allowing the generation of training data from large-scale monocular datasets, and enhancing the stereo networks' generalisation across diverse environments. The flexibility of monocular depth networks enables the synthesis of stereo-like data from various viewpoints and scenes, facilitating experimentation in stereo depth estimation. Moreover, this method adapts to different image types, including natural scenes and underwater environments, ensuring that stereo networks are trained on representative and diverse datasets. By using natural images, the synthesised stereo data retains realistic textures and layouts, improving the performance of stereo networks in real-world applications.

Algorithm 3 Stereo Pair Generation

Input: I, d, s
Output: I′
WarpedImage ← _WARP_STEREO_IMAGE(I, d, s)
I′ ← _INPAINT(WarpedImage)
return I′

1) Algorithms: Stereo pair generation is quite simple, given we have an image and its estimated inverse depth, and disparity; we first perform forward warping to shift the original image using the inverse depth, this will create our synthesised image. However, this synthesised image will have some missing information due to the shift. After generating the synthesised image using forward warping, the missing information can be filled using inpainting techniques.

We propose two ways to perform forward warping to obtain the stereo pair. In Alg. 4, the image is simply shifted using the inverse depth or disparity with some scaling factor to simulate physical camera properties such as different focal lengths and baselines.

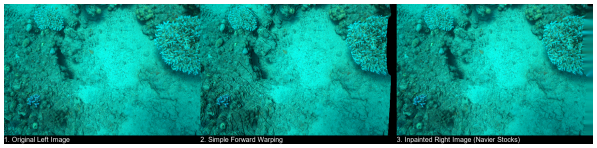

Fig. 5: Right Image Synthesis using Simple forward warping Alg. 4 and Navier Stocks inpainting Technique [20].

In Alg. 5, we expand the dimensions warped image to ensure all shifted pixels are properly accommodated within the expanded boundaries. This expansion step is crucial to prevent any loss of information during the warping process, especially when shifting pixels to simulate different viewpoints in stereo imaging. Subsequently, a cropping operation is performed on the warped image to refine its dimensions and eliminate any excess areas resulting from the expansion. The cropping process is delineated in the auxiliary function "CROP STEREO IMAGE," outlined in lines 15-20 of the algorithm. Alg. 5 aims to maximise the usable area in the original image after warping.

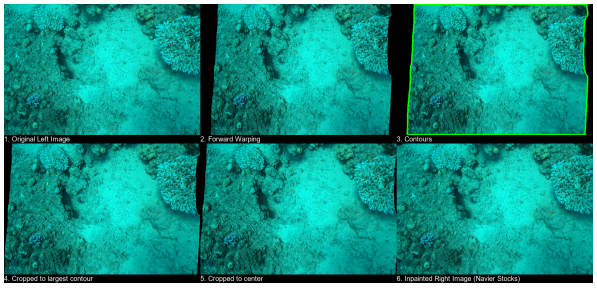

Fig. 6: Right Image Synthesis using Complex forward warping Alg. 5 and Navier Stocks inpainting Technique [20].

The cropping procedure begins by converting the warped stereo image to grayscale to simplify subsequent processing. Next, the contours of non-black regions within the grayscale image are identified. This step enables the identification of the region of interest in the expanded image. Following contour identification, the bounding box encompassing the largest contour is extracted. This bounding box is a spatial reference for extracting the region of interest within the stereo image. Subsequently, the stereo image is cropped using the extracted bounding box and then cropped again to the original image dimensions.

However, we observed simple forward warping as outlined in Alg. 4 proved more efficient and yielded better results than the complex forward warping in Alg. 5.

Algorithm 4 Simple forward Warping

Input: I, d, s
Output: I′
/*shift and store*/
for y **in** height **do**
  for x **in** width **do**
    $x' \leftarrow x-$int(d[y, x] × s)
    if $x' \geq 0$ **and** $x' <$ width **then**
      I'[y, x'] ← I[y, x]
    end if
  end for
end for
return I′

Algorithm 5 Complex forward warping

function _WARP_STEREO_IMAGE
  Input: I, d, s
  Output: I′
  Initialize I′ to zeros, with size enough to fit all shifted pixels
  max_shift ← int(max(d[y, x] × s)
  for y **in** I.height **do**
    for x **in** I.width **do**
      $x' \leftarrow x-$int(d[y, x] × s) + max_shift
      I′[y, x] ← I[y, x]
    end for
  end for
  /*Crop the image back to original size*/
  I' ← _CROP_WARPED_IMAGE(I′)
  return WarpedImage
end function
/*Helper function to crop the expanded image*/
function _CROP_WARPED_IMAGE(I)
  I′ ←Convert I to grayscale
  Find contours of non-black regions
  Extract bounding box of largest contour
  I′ ← Crop I′ using the bounding box
  I′ ← Crop I′ to center of original dimensions
  return CroppedImage
end function

2) Fine-tuning with synthetic stereo pair: When performing inference on fine-tuned models, we observed that using a constant scaling factor s for forward-warping algorithms results in artifacts. A fixed scalar s encodes a specific camera configuration during fine-tuning.

Inspired by [5], we adopt a random scaling factor $s \sim U[a, b]$. Incorporating a random scalar improves performance by making the network more robust due to exposure to diverse baselines and focal lengths during training.

We determine a and b based on camera configuration values and information loss due to forward warping. In Algorithms 4 and 5, the effective shift is defined as:

$$d' = \lfloor d \times s \rfloor$$

From Section III-D, b is limited by information loss due to forward warping. We set the maximum shift to $1/6$ of the

image width (576 pixels). Rearranging Eq. III-D.2 for $s = b$:

$$b = \frac{576}{6 \times 20} = 4.8$$

To account for extremes, we use $b = 5$.

For a, we consider typical values for focal length and baseline distance:

$$d = \frac{f_x B}{z}$$

With $z \in [0.1, 16]$ meters, $\frac{1}{z} \in [0.0625, 10]$ meters^{-1}. Focal lengths range from 1,000 to 10,000 pixels, and baselines from 0.01 to 0.5 meters. Thus:

$$0.625 \leq \frac{f_x B}{z} \leq 50,000$$

Considering extremes, we set $a = 0.5$.

Therefore, we use $s \sim U[0.5, 5]$.

E. Loss functions

1) Relative similarity loss:

$$\mathcal{L}_s = \frac{1}{HW} \sum_{i=1}^{HW} \left| \frac{d_i^S - d_i^T}{d_i^T} \right| \tag{8}$$

2) Relative trimmed similarity loss: Trimmed similarity loss is defined as

$$\mathcal{L}_{ts} = \frac{1}{T} \sum_{i \in M_{trim}} \left| \frac{d_i^S - d_i^{org}}{d_i^{org}} \right| \tag{9}$$

where M_{trim} is the set of pixels left after removing $trim_{rate}HW$ pixels with largest absolute difference and d^{org} is the original student model.

3) Bound loss: For completeness, we re-introduce the bound losses from [6]. We utilise these bound losses to fine-tune a depth of anything for underwater scenarios

$$\mathcal{L}_{bl} = \sum_{c \in \{G,B\}} \sum_{i=1}^{HW} f(-I_{c,i} + (1 - \hat{t}_{c,i}^S)\hat{B}_c^\infty)/HW \tag{10}$$

$$\mathcal{L}_{bu} = \sum_{c \in \{G,B\}} \sum_{i \in M_b} f(I_{c,i} - \hat{t}_{c,i}^T - (1 - \hat{t}_{c,i}^S)\hat{B}_c^\infty)/N_b \tag{11}$$

where, $f(x) = ReLU(x)$ and N_b is the size of set M_b.

The $\mathcal{L}_{bl}$, prevents from overestimating the depth of the distant regions. When d_i^S is overestimated, so is the medium transmission rate. It sets the lower limit on the estimated medium transmission rate $\hat{t}_{c,i}^S \geq 1 - \frac{I_{c,i}}{\hat{B}_c^\infty}$.

The $\mathcal{L}_u$ aims to set an upper bound on depth in the distant region. It assumes that the $\hat{t}_{c,i}^S$ in the background regions is greater than the true value. The background region set M_b is found by assuming $J_{c,i} \approx 0$, then

$$M_b = \left\{ i \;\middle|\; \frac{\hat{B}_c^{\infty}(1 - \hat{t}_{c,i}^T)}{I_{c,i}} \right\} \tag{12}$$

The final bound loss is a weighted sum of bound losses.

$$\mathcal{L}_b[\alpha] = \alpha_0 \mathcal{L}_{bl} + \alpha_1 \mathcal{L}_{bu} \tag{13}$$

4) Edge-aware smoothing loss: The edge-aware smoothness loss is introduced in stereo to control the high amount of artifacts due to bound losses in stereo. This usually happens due to extensive shrinkage of the estimated depth in dark regions and over-exaggeration of details. Therefore to minimise these issues we use edge-aware smoothness. Similarly to [22], the loss function is

$$\mathcal{L}_e = \sum_{c \in \{G,B\}} \sum_{i=1}^{HW} (|\partial_x d_i^*| e^{-|\partial_x I_{c,i}|} + |\partial_y d_i^*| e^{-|\partial_y I_{c,i}|})/HW \tag{14}$$

where $d^* = d^S / \bar{d^S}$ is the mean-normalised inverse depth.

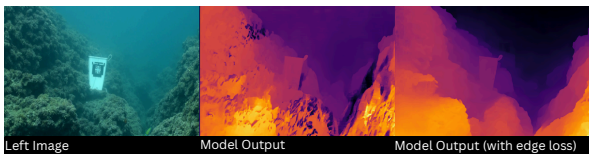

Fig. 7: Comparing effects of using high similarity loss function weights and use of edge loss to mitigate them.

F. Preparing datasets

We use monocular dataset for training and stereo dataset for evaluation. Both monocular and stereo datasets require some pre-processing. We use OpenCV [23], Pillow [24], RawPy [25] to do the pre-processing. We downsample all raw images to $(w, h) = (576, 384)$.

1) Monocular Datasets: Most underwater datasets contain raw images that are too dark, and therefore parameter estimation steps can not be performed on them. To solve this problem, we perform white balancing on the raw images. These white-balanced linear images are then used to pre-compute the underwater perimeter estimates.

Due to the nature of these white-balanced images, not all images may have good point selection, which will lead to incorrect estimation of underwater parameters for that image. We remove images where the underwater parameter estimation algorithm results in a bad estimate from the training dataset.

We also create non-linear images by performing gamma correction. These non-linear images are used as the inputs to all models mentioned in this paper.

2) Stereo Image rectification: Stereo Datasets often contain a raw set of images that need to be rectified. This process requires a set of intrinsic matrices for each camera: focal lengths, skew coefficient and principal point offsets; and extrinsic parameters: rotation matrices and baseline. These parameters are usually known and can be easily estimated when unknown. The literature is full of many techniques for both underwater and in-air camera calibration techniques which can be used to estimate intrinsic camera parameters, distortion coefficients as well as pose estimation [1,26,27]. We use OpenCV to perform the stereo rectification to remap and vertically align the image using the principle point alignment method [23].

Datasets	Type	Model	AbsRel ↓	RMSE ↓	RMSE Log ↓	SI Log ↓	$\delta < 1.05$ ↑	$\delta < 1.05^2$ ↑	$\delta < 1.05^3$ ↑	R-value↑
Canyon 1	Monocular	Depth Anything	1.523e+00	2.842e+00	5.736e-01	4.150e-01	1.081e-01	2.083e-01	3.045e-01	***9.572e-01***
		PUDE	1.194e+00	3.376e+00	6.053e-01	3.987e-01	8.769e-02	1.719e-01	2.544e-01	9.251e-01
		Ours (Underwater Depth Anything)	7.183e-01	***2.486e+00***	4.611e-01	***3.592e-01***	***1.486e-01***	***2.781e-01***	***3.934e-01***	9.454e-01
	Stereo	Unimatch	7.212e-01	2.909e+00	5.294e-01	4.531e-01	1.151e-01	2.387e-01	3.606e-01	9.244e-01
		Ours (Underwater Unimatch)	***5.157e-01***	**2.692e+00**	***4.488e-01***	**3.808e-01**	**1.385e-01**	**2.645e-01**	**3.864e-01**	9.240e-01
Canyon 2	Monocular	Depth Anything	7.845e-01	2.623e+00	5.155e-01	3.633e-01	1.046e-01	2.071e-01	2.983e-01	***9.659e-01***
		PUDE	9.132e-01	3.057e+00	5.607e-01	3.758e-01	1.100e-01	2.119e-01	2.978e-01	9.281e-01
		Ours (Underwater Depth Anything)	***5.541e-01***	***2.470e+00***	***4.208e-01***	***3.097e-01***	***1.484e-01***	***2.855e-01***	***4.034e-01***	9.573e-01
	Stereo	Unimatch	9.937e-01	3.132e+00	5.856e-01	4.757e-01	1.146e-01	2.298e-01	3.386e-01	9.285e-01
		Ours (Underwater Unimatch)	**6.357e-01**	**2.932e+00**	**4.744e-01**	**3.745e-01**	**1.288e-01**	**2.525e-01**	**3.610e-01**	9.241e-01
Rock Garden 1	Monocular	Depth Anything	7.549e+00	3.122e+00	9.101e-01	7.533e-01	1.135e-01	2.141e-01	3.081e-01	***9.520e-01***
		PUDE	2.239e+00	3.851e+00	8.399e-01	5.716e-01	8.314e-02	1.538e-01	2.148e-01	9.420e-01
		Ours (Underwater Depth Anything)	5.252e+00	***2.945e+00***	8.090e-01	6.912e-01	1.275e-01	2.465e-01	3.495e-01	9.445e-01
	Stereo	Unimatch	1.489e+00	3.175e+00	8.082e-01	6.673e-01	1.233e-01	2.505e-01	3.592e-01	9.406e-01
		Ours (Underwater Unimatch)	***1.290e+00***	**3.023e+00**	***6.866e-01***	***5.617e-01***	***1.307e-01***	***2.632e-01***	***3.775e-01***	9.392e-01
Rock Garden 2	Monocular	Depth Anything	9.353e-01	2.658e+00	5.405e-01	3.934e-01	1.171e-01	2.261e-01	3.271e-01	***9.660e-01***
		PUDE	1.298e+00	4.091e+00	7.424e-01	4.572e-01	5.933e-02	1.214e-01	1.843e-01	9.084e-01
		Ours (Underwater Depth Anything)	6.425e-01	2.681e+00	4.788e-01	3.553e-01	1.202e-01	2.337e-01	3.355e-01	9.391e-01
	Stereo	Unimatch	5.443e-01	2.375e+00	4.385e-01	3.854e-01	***1.814e-01***	***3.473e-01***	***4.833e-01***	9.317e-01
		Ours (Underwater Unimatch)	***4.698e-01***	***2.369e+00***	***4.072e-01***	***3.477e-01***	1.798e-01	3.439e-01	4.793e-01	9.277e-01

TABLE I: **Quantitative Results** on the FLSea datasets. The New underwater stereo model outperforms other models in almost all metrics. The best results are bolded and italicized. Underwater Unimatch results are coded into 3 colors, green when it performs the best compared to all models, blue when it performs the best only to stereo models and no color if neither.

3) Ground truth depth to Disparity: Both parent and daughter networks predict inverse depth, i.e. disparity. Therefore we need to convert the ground truth depth z to disparity d.

$$d = \frac{f_x B}{z}$$

We then rescale the disparity to be between $[0, 20]$, to keep it consistent with the network outputs.

4) Stereo dataset ground truth filtering: Ground truth in the stereo datasets often has regions where the depth couldn't be resolved. The estimates next to the unresolved regions are usually highly uncertain and need to be removed.

The re-scaling of ground truth disparity would not be correct due to these highly uncertain values. This effect can be seen in Fig. 8.

To tackle this issue we introduce a general approach to filter outlier points. We split this into two steps: i. statistics-based filtering: removing highly unlikely points through IQR and z-score thresholding, and ii. binary mask filtering: removing the highly uncertain points next to unresolved depth regions.

i. Statistics-based filtering: We set our lower fences for IQR thresholding at $1.5IQR$, i.e. any points above $Q_3 + 1.5IQR$ or below $Q_1 - 1.5IQR$ are considered an outlier. For z-score thresholding, we consider all points with $|\text{z-score}| > 3$

Fig. 8: Scaled disparity from the ground truth without any filtering

are also considered an outlier. We remove all points classified as outliers with any of these methods.

ii. Binary mask filtering: Unresolved regions are set to a value of 0, therefore we take advantage of this fact and create a binary mask of the unresolved region. This binary mask is then dilated using a kernel size of 3 and a dilation iteration of 2. From Fig. 9, we can see that this dilated mask will target a very thin edge in the resolved depth regions, removing highly uncertain points right next to the unresolved regions.

It is worth noting from Fig. 10 and 11, that we are not able to remove all outliers if we apply only one of the methods. Therefore both filtering techniques are required to remove the highly uncertain points.

Fig. 9: Binary masks to remove the highly uncertain points on the boundaries.

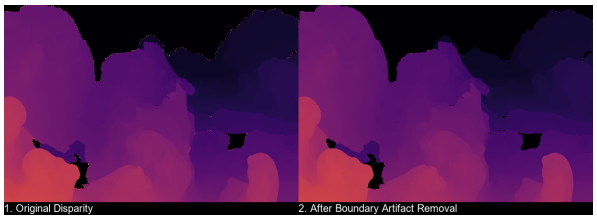

Fig. 10: 1. Original Disparity without unscaling. 2. Scaled Disparity after only binary mask filtering.

After applying both filtering, from Fig. 11 we see that we can remove most of these artifacts and successfully scale the disparity.

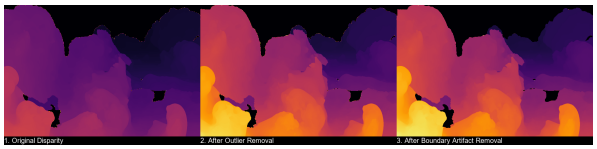

Fig. 11: 1. Original Disparity without unscaling. 2. Disparity after statistics-based filtering. 3. Scaled Disparity after binary mask filtering.

IV. Experiments

A. Implementation details

The overall pipeline can be separated into three parts, 1. Estimate the underwater parameters on the complete combined dataset. 2. Finetune the monocular model using PUDE framework. 3. Finetune the stereo model using synthetic image pairs generated using monocular model.

For training, we create a combined dataset from Sea-Thru [28] and SeaThru-Nerf [29]. We remove the images with bad parameter estimation as outlined in section III-F. After filtering, the combined dataset has 112 images. We randomly shuffle the combined dataset to ensure even exposure to different environments.

We perform a zero-shot evaluation on the FLSea dataset [1]. In particular, we infer the left images and use their corresponding ground truth for evaluation.

1) Monocular model fine tuning: We initialise both Teacher and Student models using depth anything, looking to further fine-tune it for underwater scenarios using the PUDE training framework. We use the Adam optimiser [30] and decay the learning rate with an initial learning rate of $1e-6$ and a linear schedule with a step size of 1 and a learning rate decay factor of 0.1. The fine-tuning is performed over 3 epochs.

The total loss function for monocular can be taken as the weighted sum of trimmed similarity and bound loss functions, i.e.

$$\mathcal{L}_{total}[\beta] = \beta_0\mathcal{L}_{ts} + \beta_1\mathcal{L}_b[\alpha], \tag{15}$$

where, $\beta = [1, 0.1], \alpha = [10, 0.4], trim_{rate} = 0.3$.

2) Stereo model fine-tuning: We initialise the student stereo model using Unimatch and the monocular teacher model using the fine-tuned Depth Anything. We use AdamW optimiser [31] with an initial learning rate of $1e-5$ and decay the learning rate with a cosine learning rate scheduler similar to the Unimatch training loop [9].

The loss function is a weighted sum of similarity loss, the edge-aware smoothness loss and the trimmed similarity loss.

$$\mathcal{L}_{total}[\beta] = \beta_0\mathcal{L}_s + \beta_1\mathcal{L}_e + \beta_2\mathcal{L}_{ts}, \tag{16}$$

where, $\beta = [2, 0.1, 0.5], trim_{rate} = 0.3$.

B. Evaluation metrics

We utilise the widely used metrics in the depth estimation literature. For completeness, these are:
Define $\mathbb{E}(x) := \frac{1}{K}\sum_{i=1}^{K} x$, then;
Absolute relative error,

$$\text{AbsRel}(d, \hat{d}) = \mathbb{E}\left(\frac{|\hat{d} - d|}{d}\right)$$

Root mean squared error,

$$\text{RMSE}(d, \hat{d}) = \sqrt{\mathbb{E}(\hat{d}_i - d_i)^2}$$

Root mean square logarithmic error,

$$\text{RMSE}\log(d, \hat{d}) = \sqrt{\mathbb{E}(e^2)}, e = \log(\hat{d}) - \log(d)$$

Scale-invariant logarithmic error,

$$\text{SI}\log(d, \hat{d}) = \mathbb{E}(e^2) - (\mathbb{E}(e))^2$$

Accuracy with threshold(γ),

$$\delta(d, \hat{d}) = \max(k_i, k_i^{-1}) < \gamma, k_i = \frac{\hat{d}_i}{d_i}$$

We are also interested in the linearity between the predicted inverse depth d and the inverse ground truth $1/z$, therefore we also consider Pearson's correlation coefficient as one of the metrics [32].

C. Zero-shot evaluations

1) Quantitative evaluation: The models are compared against each other based on the evaluation metrics described above. The inference time for SDE models was observed to be considerably greater than the MDE model and should be considered in future studies. We compare the monocular and stereo models on the FLSea dataset, with the results presented in Table I. The table shows that the underwater Unimatch model performs better than other models on most metrics. It also demonstrates comparable performance to the teacher model, Underwater Depth Anything.

However, the underwater Unimatch model performed worse on the linearity metric. This was unexpected, as we anticipated an increase in linearity with improved model accuracy after

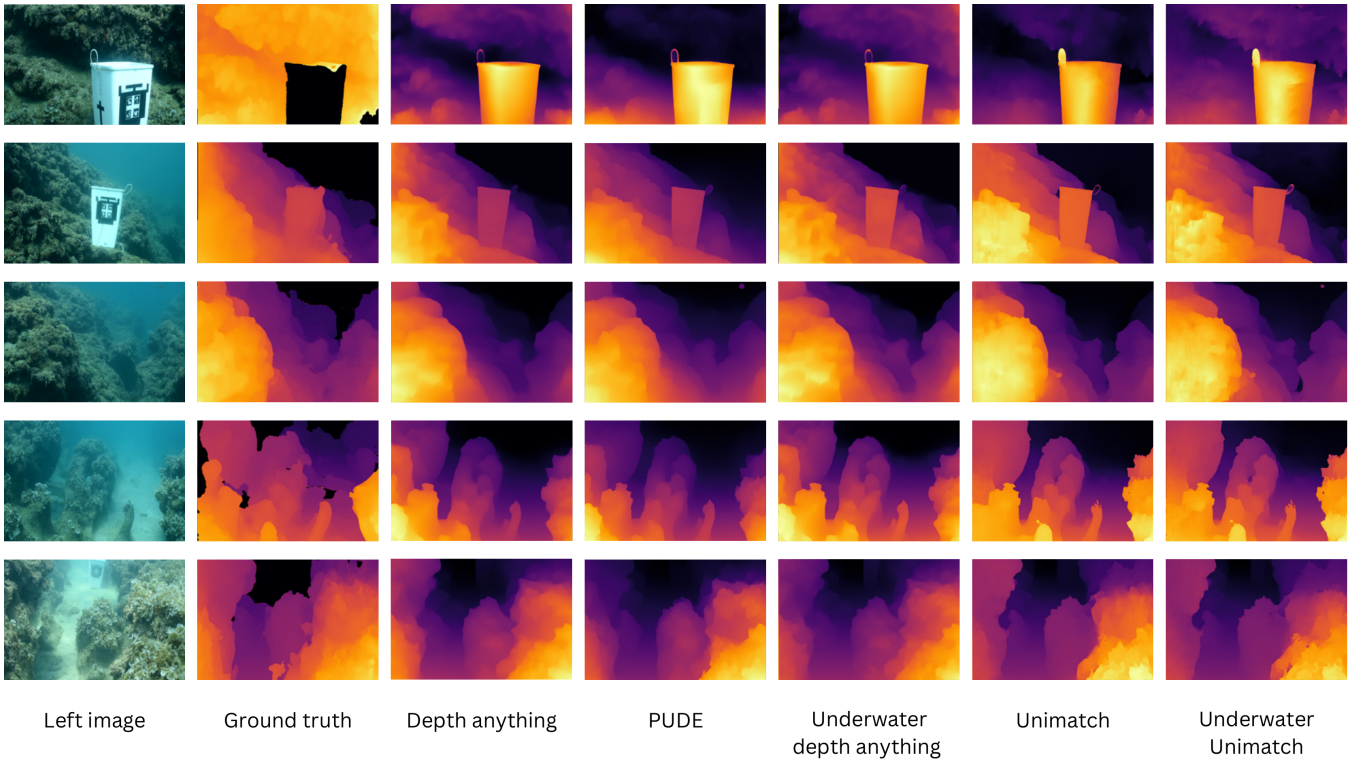

Fig. 12: **Qualitative results** on the FLSea datasets. Underwater Unimatch shows consistently sharper and more detailed estimations across all datasets in FLSea.

fine-tuning. This pattern of a decrease in the r-value after fine-tuning was also observed in the MDE case. Notably, the r-value of the Underwater Depth Anything model is consistently less than that of the Depth Anything model across all datasets. These differences might be explained by the increased detail in the underwater Unimatch's inverse depth estimations compared to the ground truth. Canyon 2 was the only dataset where the underwater Unimatch outperformed only stereo models, while the monocular model, Underwater Depth Anything, outperformed all models across all metrics in this dataset.

In the Rock Garden 2 dataset, the Underwater Unimatch model struggles with the accuracy metrics. By looking at the qualitative results in Fig. 12, we observe that the underwater Unimatch estimates tend to be much sharper than those of other models and the ground truth. This increase in resolution could explain its slightly lower performance compared to the original Unimatch model on accuracy metrics for this dataset.

2) Qualitative evaluation: We visualized our models on four datasets from FLSea, comparing and contrasting their qualitative performance based on two criteria: increase in detail and correction of depth estimates in nearby regions. The qualitative results are presented in Fig. 12. We observed that both Underwater Depth Anything (MDE) and Underwater Unimatch (SDE) models showed an increase in detail, particularly in nearby regions.

Moreover, the Underwater Unimatch's predictions remained true to the silhouette of objects, providing much sharper outputs. This is particularly evident in the third row of Fig. 12, where the silhouette of a fish is well-traced by Underwater Unimatch, unlike other models. This improvement is likely due to the introduction of the edge-aware smoothing loss. Additionally, the Underwater Unimatch model proved to be more robust with very close objects while retaining detail in the background, as shown in the first row of Fig. 12.

D. Ablation studies

We compare the effects of all three loss functions through ablation studies. The training setup is similar to Sec. IV-A.2. Table II shows that all three loss functions contribute to the training. We observe the largest improvement when all loss functions are used together.

An interesting observation can be made when the similarity and trimmed similarity losses are used together. We see that this combination results in a decrease in overall accuracy metrics. Fig. 13 gives us more insights into this. The $\mathcal{L}_s, \mathcal{L}_{ts}$ combination results in better estimation of the background region than just $\mathcal{L}_s$ when compared to the ground truth. The combination of $\mathcal{L}_s, \mathcal{L}_{ts}, \mathcal{L}_e$ works the best as the $\mathcal{L}_e$ pushes the model to ensure depth estimation conforms well to the edge features of the left input image.

By linking these observations with our earlier evaluations, we can gain a deeper understanding of the model's performance. The quantitative evaluation revealed that the underwater Unimatch model performs well on most metrics but has some unexpected results in linearity and specific datasets. The qualitative evaluation showed that the model provides sharper object silhouettes and robustness with very close objects. The ablation studies further clarify that the edge-aware smoothing loss ($\mathcal{L}_e$) plays a crucial role in enhancing

$\mathcal{L}_s$	$\mathcal{L}_{ts}$	$\mathcal{L}_e$	AbsRel ↓	RMSE ↓	RMSE Log ↓	SI Log ↓	$\delta < 1.05$ ↑	$\delta < 1.05^2$ ↑	$\delta < 1.05^3$ ↑	R-value↑
✓			6.792e-01	2.744e+00	5.191e-01	4.271e-01	1.292e-01	2.543e-01	3.733e-01	***9.269e-01***
✓	✓		6.163e-01	2.904e+00	4.989e-01	4.208e-01	1.209e-01	2.438e-01	3.533e-01	9.124e-01
✓	✓	✓	***5.157e-01***	***2.692e+00***	***4.488e-01***	***3.808e-01***	***1.385e-01***	***2.645e-01***	***3.864e-01***	9.240e-01

TABLE II: **Ablation study** on the Canyon 1 dataset. The New underwater stereo model outperforms other models in almost all metrics. The best results are bolded and italicised. The highlighted green cells identify when a model with all three loss functions performs the best compared to all models.

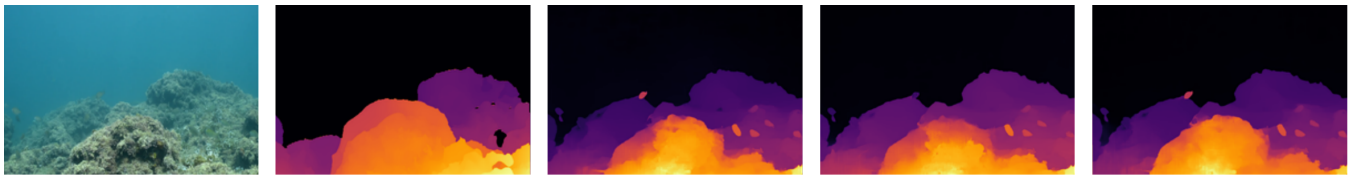

Fig. 13: **Qualitative results** of ablation study on a random image from Canyon 1 dataset.

the model's performance by improving the conformity of depth estimation to edge features. This comprehensive analysis highlights the strengths and areas for improvement in the underwater Unimatch model, providing a clear direction for future enhancements.

V. Conclusion and Future Work

This work demonstrated that in-air monocular and stereo-depth estimation models could be effectively combined to create a self-supervised training loop for underwater stereo-depth estimation via transfer learning. To address the issue of the lack of stereo-pair data, we illustrated how a synthetic stereo-pair dataset can be generated using a monocular dataset and a well-trained underwater model. Our quantitative and qualitative analyses showed that this synthetic dataset could be effectively used to enhance the overall performance of the underwater stereo model.

There are several avenues for future work. One approach could involve using a mix of synthetic and real datasets to fine-tune the model, rather than relying solely on synthetic data. This would increase the amount of training data available and allow the model to learn real textures, as opposed to the blurry infilling produced by the Navier-Stokes infilling technique. Additionally, text-to-image infilling deep learning models could be explored to generate more realistic textures for unknown areas, further improving the lack of textures and accuracy of the dataset. Previous work by [2] also showed that using synthetic data with strong perturbations challenges the model to learn more robust representations. Therefore, the effects of using strong perturbations on the synthetic stereo dataset, such as color distortions, Gaussian blurring, and spatial distortions, should be studied in detail.

Furthermore, another possible avenue to increase underwater performance is to use underwater stereo models to fine-tune in-air monocular models for underwater scenarios. Monocular models were observed to have faster inference time and, therefore, are more practical than stereo models utilized in this study. In this paper, we showed that the fine-tuned underwater SDE model has more detail and produces much sharper results. Therefore, this knowledge can be transferred to the monocular models through a similar framework illustrated in Figure 1.

References

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

[2] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, "Depth anything: Unleashing the power of large-scale unlabeled data," 2024.

[3] Y. Wu, Y. Zhou, S. Chen, Y. Ma, and Q. Li, "Defect inspection for underwater structures based on line-structured light and binocular vision," *Appl. Opt.*, vol. 60, no. 25, pp. 7754–7764, Sep 2021. [Online]. Available: https://opg.optica.org/ao/abstract.cfm?URI=ao-60-25-7754

[4] A. Filisetti, A. Marouchos, A. Martini, T. Martin, and S. Collings, "Developments and applications of underwater lidar systems in support of marine science," in *OCEANS 2018 MTS/IEEE Charleston*, 2018, pp. 1–10.

[5] J. Watson, O. M. Aodha, D. Turmukhambetov, G. J. Brostow, and M. Firman, "Learning stereo from single images," 2020.

[6] J. Yang, M. Gong, and Y. Pu, "Physics-informed knowledge transfer for underwater monocular depth estimation," 2024, under review.

[7] J. Y. Chiang and Y.-C. Chen, "Underwater image enhancement by wavelength compensation and dehazing," *IEEE Transactions on Image Processing*, vol. 21, no. 4, pp. 1756–1769, 2012.

[8] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 12 179–12 188.

[9] H. Xu, J. Zhang, J. Cai, H. Rezatofighi, F. Yu, D. Tao, and A. Geiger, "Unifying flow, stereo and depth estimation," 2023.

[10] M. Bertalmio, L. Vese, G. Sapiro, and S. Osher, "Simultaneous structure and texture image inpainting," *IEEE Transactions on Image Processing*, vol. 12, no. 8, pp. 882–889, 2003.

[11] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00. USA: ACM Press/Addison-Wesley Publishing Co., 2000, p. 417–424. [Online]. Available: https://doi.org/10.1145/344779.344972

[12] J. Shen, S. H. Kang, and T. F. Chan, "Euler's elastica and curvature-based inpainting," *SIAM Journal on Applied Mathematics*, vol. 63, no. 2, pp. 564–592, 2003. [Online]. Available: https://doi.org/10.1137/S0036139901390088

[13] A. Tsai, A. Yezzi, and A. Willsky, "Curve evolution implementation of the mumford-shah functional for image segmentation, denoising, interpolation, and magnification," *IEEE Transactions on Image Processing*, vol. 10, no. 8, pp. 1169–1186, 2001.

[14] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Trans. Graph.*, vol. 36, no. 4, jul 2017. [Online]. Available: https://doi.org/10.1145/3072959.3073659

[15] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[16] A. Bugeau, M. Bertalmío, V. Caselles, and G. Sapiro, "A comprehensive framework for image inpainting," *IEEE Transactions on Image Processing*, vol. 19, no. 10, pp. 2634–2645, 2010.

[17] T. Yu, Z. Guo, X. Jin, S. Wu, Z. Chen, W. Li, Z. Zhang, and S. Liu, "Region normalization for image inpainting," 2023.

[18] H. Liu, Z. Wan, W. Huang, Y. Song, X. Han, and J. Liao, "Pd-gan: Probabilistic diverse gan for image inpainting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2021, pp. 9371–9381.

[19] Y. Zeng, J. Fu, H. Chao, and B. Guo, "Aggregated contextual transformations for high-resolution image inpainting," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 7, pp. 3266–3280, 2023.

[20] M. Bertalmio, A. Bertozzi, and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–I.

[21] P. L. Drews, E. R. Nascimento, S. S. Botelho, and M. F. Montenegro Campos, "Underwater depth estimation and image restoration based on single images," *IEEE Computer Graphics and Applications*, vol. 36, no. 2, pp. 24–35, 2016.

[22] C. Godard, O. M. Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," 2019.

[23] OpenCV Library, "OpenCV (Open Source Computer Vision Library)," 2024. [Online]. Available: https://opencv.org/

[24] PIL Fork Contributors, "Pillow (pil fork)," 2024. [Online]. Available: https://pillow.readthedocs.io/en/stable/

[25] LibRaw Team, "Rawpy: Python library for working with raw images," 2024. [Online]. Available: https://github.com/LibRaw/LibRaw

[26] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[27] M. Singh, M. Dharmadhikari, and K. Alexis, "An online self-calibrating refractive camera model with application to underwater odometry," 2023.

[28] D. Akkaynak and T. Treibitz, "Sea-thru: A method for removing water from underwater images," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 1682–1691.

[29] D. Levy, A. Peleg, N. Pearl, D. Rosenbaum, D. Akkaynak, S. Korman, and T. Treibitz, "Seathru-nerf: Neural radiance fields in scattering media," 2023.

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[31] I. Loshchilov and F. Hutter, "Fixing weight decay regularization in adam," *CoRR*, vol. abs/1711.05101, 2017. [Online]. Available: http://arxiv.org/abs/1711.05101

[32] D. S. Moore, G. P. McCabe, and B. A. Craig, *Introduction to the Practice of Statistics*, 9th ed. W. H. Freeman, 2017.