Learning to Remove Refractive Distortions from Underwater Images Supplementary Material

Simron Thapa Nianyi Li Jinwei Ye
Louisiana State University, Baton Rouge, LA 70803, USA
{sthapa5, nli5, jinweiye}@lsu.edu

This supplementary document consists of three sections. In Section 1, we provide detailed information on our network structure. In Section 2, we provide details on creating the synthetic dataset. Specifically, we provide the wave equations for simulating the underwater images and describe how the distortion level is quantified. In Section 3, we provide more details on our experimentation. Firstly, we provide the equations for computing the error metrics. Secondly, we show additional visual results of our ablation studies. Thirdly, we then show the visual comparisons on various testing datasets. And lastly, we show additional results for the "in-the-wild" experiments.

1. Network Architecture

Here we provide the detailed architecture of our network. Our *distortion guided network (DG-Net)* consists of two subnets: 1) a convolutional network for estimating the refractive distortion from 3 consecutive refracted images (Dis-Net) and 2) a distortion-guided GAN for generating the undistorted underwater image (DG-GAN). Table 1 and Table 2 provide detailed network architecture of the two subnets. In the tables, Conv, Deconv and BN refer to convolution layers, transpose convolution layers, and batch normalization.

	Input	Filters	Output Shape
Encoder	Input		$128 \times 128 \times 3$
	Conv+Conv, ReLU	$18@2 \times 2 \times 6$	$128 \times 128 \times 18$
	Maxpool	Stride = 2	$64 \times 64 \times 18$
	Conv+Conv, ReLU	$36@2 \times 2 \times 18$	$64 \times 64 \times 36$
	Maxpool	Stride = 2	$32 \times 32 \times 36$
	Conv+Conv+Conv, ReLU	$72@2 \times 2 \times 36$	$32 \times 32 \times 72$
	Maxpool	Stride = 2	$16 \times 16 \times 72$
	Conv+Conv+Conv, ReLU	$144@2 \times 2 \times 72$	$16 \times 16 \times 144$
	Maxpool	Stride = 2	$8 \times 8 \times 144$
	Conv+Conv+Conv, ReLU	$144@2 \times 2 \times 144$	$8 \times 8 \times 144$
	Maxpool	Stride = 2	$4 \times 4 \times 144$
Decoder	Deconv+Deconv	$72@4 \times 4 \times 144$	$8 \times 8 \times 72$
	Output_5		$8 \times 8 \times 3$
	Concat+Deconv+Deconv	$36@6 \times 6 \times 72$	$16 \times 16 \times 36$
	Output_4		$16 \times 16 \times 3$
	Concat+Deconv+Deconv	$18@4 \times 4 \times 36$	$32 \times 32 \times 18$
	Output_3		$32 \times 32 \times 3$
	Concat+Deconv+Deconv	$9@4 \times 4 \times 18$	$64 \times 64 \times 9$
	Output_2		$64 \times 64 \times 3$
	Concat+Deconv+Deconv	$3@3 \times 3 \times 9$	$128 \times 128 \times 3$
	Output_1		$128 \times 128 \times 3$
RNN	Input		$3 \times 128 \times 128 \times 3$
	convLSTM, BN	$3@2 \times 2 \times 3$	$3 \times 128 \times 128 \times 3$
	convLSTM, BN	$3@2 \times 2 \times 3$	$128 \times 128 \times 3$
	Output		$128 \times 128 \times 3$

Table 1. Detailed architecture of the Dis-Net.

	Input	Filters	Output Shape
Generator	Input		$128 \times 128 \times 3$
	Conv, Stride = 2, BN, LeakyReLU	$64@4 \times 4$	$64 \times 64 \times 64$
	Conv, Stride = 2, BN, LeakyReLU	$128@4 \times 4$	$32 \times 32 \times 128$
	Conv, Stride = 2, BN, LeakyReLU	$256@4 \times 4$	$16 \times 16 \times 256$
	Conv, Stride = 2, BN, LeakyReLU	$512@4 \times 4$	$8 \times 8 \times 512$
	Conv, Stride = 2, BN, LeakyReLU	$512@4 \times 4$	$4 \times 4 \times 512$
	Conv, Stride = 2, BN, LeakyReLU	$512@4 \times 4$	$2 \times 2 \times 512$
	Conv, Stride = 2	$512@4 \times 4$	$1 \times 1 \times 512$
	Concat+Deconv, Stride = 2, BN, ReLU	$512@4 \times 4$	$2 \times 2 \times 512$
	Concat+Deconv, Stride = 2, BN, ReLU	$512@4 \times 4$	$4 \times 4 \times 512$
	Concat+Deconv, Stride = 2, BN, ReLU	$512@4 \times 4$	$8 \times 8 \times 512$
	Concat+Deconv, Stride = 2, BN, ReLU	$256@4 \times 4$	$16 \times 16 \times 256$
	Concat+Deconv, Stride = 2, BN, ReLU	$128@4 \times 4$	$32 \times 32 \times 128$
	Concat+Deconv, Stride = 2, BN, ReLU	$64@4 \times 4$	$64 \times 64 \times 64$
	Deconv, Stride = 2	$3@4 \times 4$	$128 \times 128 \times 3$
	Output_1		$128 \times 128 \times 3$
Discriminator	Input_1		$128 \times 128 \times 3$
	Input_2		$128 \times 128 \times 3$
	Concat		$128 \times 128 \times 6$
	Conv, Stride = 2, BN, ReLU	$64@4 \times 4$	$64 \times 64 \times 64$
	Conv, Stride = 2, BN, ReLU	$128@4 \times 4$	$32 \times 32 \times 128$
	Conv, Stride = 2, BN, ReLU	$256@4 \times 4$	$16 \times 16 \times 256$
	Conv, Stride = 2, BN, ReLU	$512@4 \times 4$	$8 \times 8 \times 512$
	Conv, BN, ReLU	$512@4 \times 4$	$8 \times 8 \times 512$
	Conv	$1@4 \times 4$	$8 \times 8 \times 512$
	Activation+Output		1

Table 2. Detailed architecture of the DG-GAN

2. Synthetic Underwater Image Dataset

In this section, we show samples from our synthetic underwater image dataset. Our dataset contains around 63k distorted refraction images, generated from 6354 unique reference pattern. Most of the reference patterns are selected from the Describable Textures Dataset (DTD) [1]. Except that, we include additional $\sim$ 500 various texts images. We render 10 consecutive frames per wave. For each refraction image, we provide the ground truth distortion-free image (the reference pattern), the ground truth distortion map, and the ground truth height map of the wave. In Fig. 1, we show sample data from our dataset. The three consecutive frames are used as input to our algorithm.

2.1. Water Wave Simulation

We use the physics-based dynamic water wave simulation software presented in [8]. To show that our method is robust to different types of water turbulence, we simulate three types of waves: ripple waves, ocean waves, and Gaussian waves. In the following, we describe the simulation equations used for each type of wave.

Ripple waves. We create water ripples with damping effects. Let z_{ri} be the fluid surface height, the equation can be written as

$$z_{ri} = A \exp(-\frac{x^2 + y^2}{2\sigma^2}) \sin(\omega t) \tag{1}$$

where A is the wave amplitude, σ is the damping factor, and ω is the phase factor.

Ocean waves. The Grestner's wave equations are are widely used in computer graphics to simulate ocean waves [8, 7]. We use them to model fluid with relatively large volumes. In our implementation, we compute the Grestner's equation with its Fast Fourier Transform (FFT) form. The FFT-based representation of the equation can be written as

$$z_{gr} = \sum_m \sum_n \tilde{z}_{gr} \exp(j2\pi(mx + ny)) \tag{2}$$

where $\tilde{z}_{gr}$ is the Fourier amplitude, j is the imaginary unit, m and n are integers bounded by $[-M/2, M/2]$ and $[-N/2, N/2]$ (M and N are the dimensions of the mesh grid). We use the Phillips spectrum [7] as our height amplitude Fourier component ($\tilde{z}_{gr}$) that determines the structure of the fluid surface.

Gaussian waves. For Gaussian waves, we assume that the maximum water surface fluctuation is small compared to the height (h_0) of the fluid in the stable condition. This fluctuating water surface is governed by the wave equation:

$$z_{ga}(t+1) = 2 \times z_{ga}(t) + c^2 \nabla - z_{ga}(t-1) \tag{3}$$

Note that $z_{ga}(t=0) = 0$ and ∇ is a Laplacian operator related to $z_{ga}(t)$. Here, $c = \sqrt{gh_0}$ is the speed of the wave (g is the gravity).

Fig. 1 shows exemplary water turbulence images of different types of waves. We also show the height maps and distortion maps, which are highly relevant to the water wave fronts.

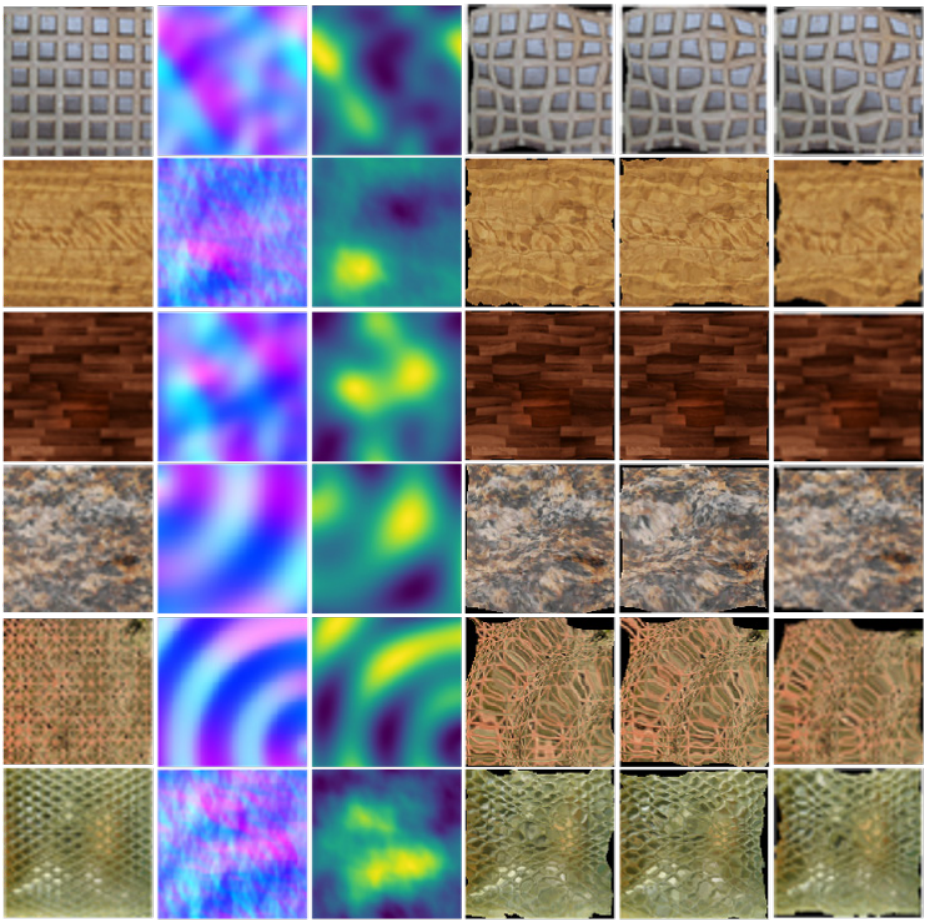

Figure 1. Sample images of our synthetic underwater image dataset. From left to right, we show the ground truth (GT) distortion-free image, GT distortion map, GT height map, and three consecutive distorted images.

2.2. Distortion Levels

In order to evaluate the robustness of our method with respect to the strength of distortion, we categorize our underwater images into seven distortion levels according to their distortion maps. Specifically, we quantify the distortion levels using the averaged magnitude of the distortion map. Given a distortion $W = \{\mathbf{w}_i\}_{i=1}^{M}$ (where $\mathbf{w}_i \in \mathbb{R}^2$ is per-pixel distortion vector and M is the total number of pixels), the distortion level d_l of its corresponding refraction image is:

$$d_l = \lceil \frac{1}{M} \sum_{i=1}^{M} \|\mathbf{w}_i\|_2 \rceil \tag{4}$$

where $\lceil \cdot \rceil$ is the ceiling operator. In our dataset, $d_l = 7$ is the largest distortion level ($d_l = 0$ indicates distortion-free). We show exemplary images in selected distortion level in Fig. 2.

Figure 2. We show three exemplary images of different distortions levels (Top) and the corrected output by our method (Bottom).

3. Additional Details on Experimentation

3.1. Evaluation Metrics

We use four standard image quality or similarity metrics for quantitative evaluation: 1) Peak Signal-to-Noise Ratio (PSNR) [3], 2) Structural Similarity Index (SSIM) [2], 3) Sum Squared Difference (SSD) [6], and 4) SSD in Gradient (SSDG) [6]. Let Y be the ground truth image and $\hat{Y}$ be our estimated image. The total number of pixels is M for both Y an $\hat{Y}$. We use Y as reference to evaluate the quality of $\hat{Y}$.

PSNR. The PSNR computes the ratio between the maximum possible value (or power) of a signal and the power of corrupting noise that affects the quality of its representation. It is computed as:

$$\text{PSNR} = 10 \log_{10} \left(\frac{R^2}{\frac{1}{M} \sum_{p} (Y(p) - \hat{Y}(p))^2} \right) \tag{5}$$

where $p \in [1, M]$ is the pixel index and R is the maximum possible pixel value of image (*e.g.*, when using an 8-bit image, $R = 255$).

SSIM. The SSIM is a perceptual metric that is used to quantify the image quality. It assesses the visual impact of three key features of an image: luminance, contrast, and structure. It is computed as a multiplicative combination of the three terms:

$$\text{SSIM} = [l(Y, \hat{Y})]^{\alpha} \cdot [c(Y, \hat{Y})]^{\beta} \cdot [s(Y, \hat{Y})]^{\gamma} \tag{6}$$

where $l(Y, \hat{Y}) = \frac{2\mu_Y \mu_{\hat{Y}} + C_1}{\mu_Y^2 \mu_{\hat{Y}}^2 + C_1}$, $c(Y, \hat{Y}) = \frac{2\sigma_Y \sigma_{\hat{Y}} + C_1}{\sigma_Y^2 \sigma_{\hat{Y}}^2 + C_2}$, and $s(Y, \hat{Y}) = \frac{\sigma_{Y\hat{Y}} + C_3}{\sigma_Y \sigma_{\hat{Y}} + C_3}$ represent the luminance, contrast, and structure terms, respectively. μ_Y, $\mu_{\hat{Y}}$, σ_Y, $\sigma_{\hat{Y}}$, and $\sigma_{Y\hat{Y}}$ are the local means, standard deviations, and cross-covariance for

images Y, $\hat{Y}$. When $\alpha = \beta = \gamma = 1$ and $C3 = C2/2$, the index simplifies to:

$$\text{SSIM} = \frac{(2\mu_Y \mu_{\hat{Y}} + C_1)(2\sigma_{Y\hat{Y}} + C_2)}{(\mu_Y^2 \mu_{\hat{Y}}^2 + C_1)(\sigma_Y^2 \sigma_{\hat{Y}}^2 + C_2)} \tag{7}$$

SSD. The SSD evaluates the per-pixel difference between $\hat{Y}$ and Y. It is computed as:

$$\text{SSD}(Y, \hat{Y}) = \frac{1}{M} \sum_p [Y(p) - \hat{Y}(p)]^2 \tag{8}$$

where $p \in [1, M]$ is the pixel index.

SSDG. The SSDG computes the SSD in terms of the intensity gradients. It is computed as:

$$\text{SSDG}(Y, \hat{Y}) = \text{SSD}(Y_x, \hat{Y}_x) + \text{SSD}(Y_y, \hat{Y}_y) \tag{9}$$

where Y_x, $\hat{Y}_x$, Y_y, and $\hat{Y}_y$ are the horizontal and vertical gradients for Y and $\hat{Y}$, respectively.

3.2. Visual Results of Ablation Studies

Fig. 3 we show the visual results of effect of the physical constrains. Specifically, we compare our full network (DG-Net) with the Dis-Net (without the distortion-guided GAN), and three variants of the Dis-Net: 1) Dis-Net$_W$, which removes the last two terms of $\mathcal{L}_W$ (notice that these terms are constrained by our physical model); 2) Dis-Net$_R$, which removes the refraction loss $\mathcal{L}_R$ in Dis-Net; and 3) Dis-Net$_C$, which removes the consistency loss $\mathcal{L}_C$ in Dis-Net. We can see that all loss terms contribute to improve our network performance.

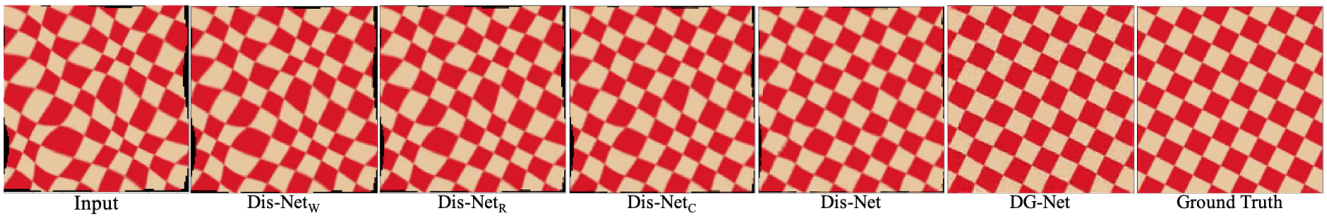

Figure 3. Qualitative ablation on physics-based loss terms. We compare the results from different ablative sub-networks to the ground truth.

3.3. Visual Comparative Results

Fig. 4 shows the visual comparion of our method with Li *et al*. [5] in our synthetic dataset (SynSet). We compare both the estimated distortion map and the distortion-free images. We can see that our method achieves better accuracy on both as we account for the temporal consistency by three images as input. Notice that the model-based methods [9] and [6] cannot be applied on the SynSet as they need long input sequence.

Fig. 5 shows the visual comparisons on Tian's real dataset "Brick", "Small", and "Tiny" [9]. We compare with Tian *et al*. [9], Oreifej *et al*. [6], Li *et al*. [5], and James *et al*. [4]. Fig. 6 shows the qualitative comparison with all the above methods on ThapaSet [8].

We see that our method achieves better accuracy and visually pleasing results with shorter input sequence of 3 frames when compared with 10 frames input to Tian *et al*. [9], Oreifej *et al*. [6], James *et al*. [4] and single input to Li *et al*. [5]. Here in Fig. 7, we further compares our method with Tian *et al*. [9], Oreifej *et al*. [6], James *et al*. [4] when 61 frames are taken as input. We see that our method can still obtain comparable results.

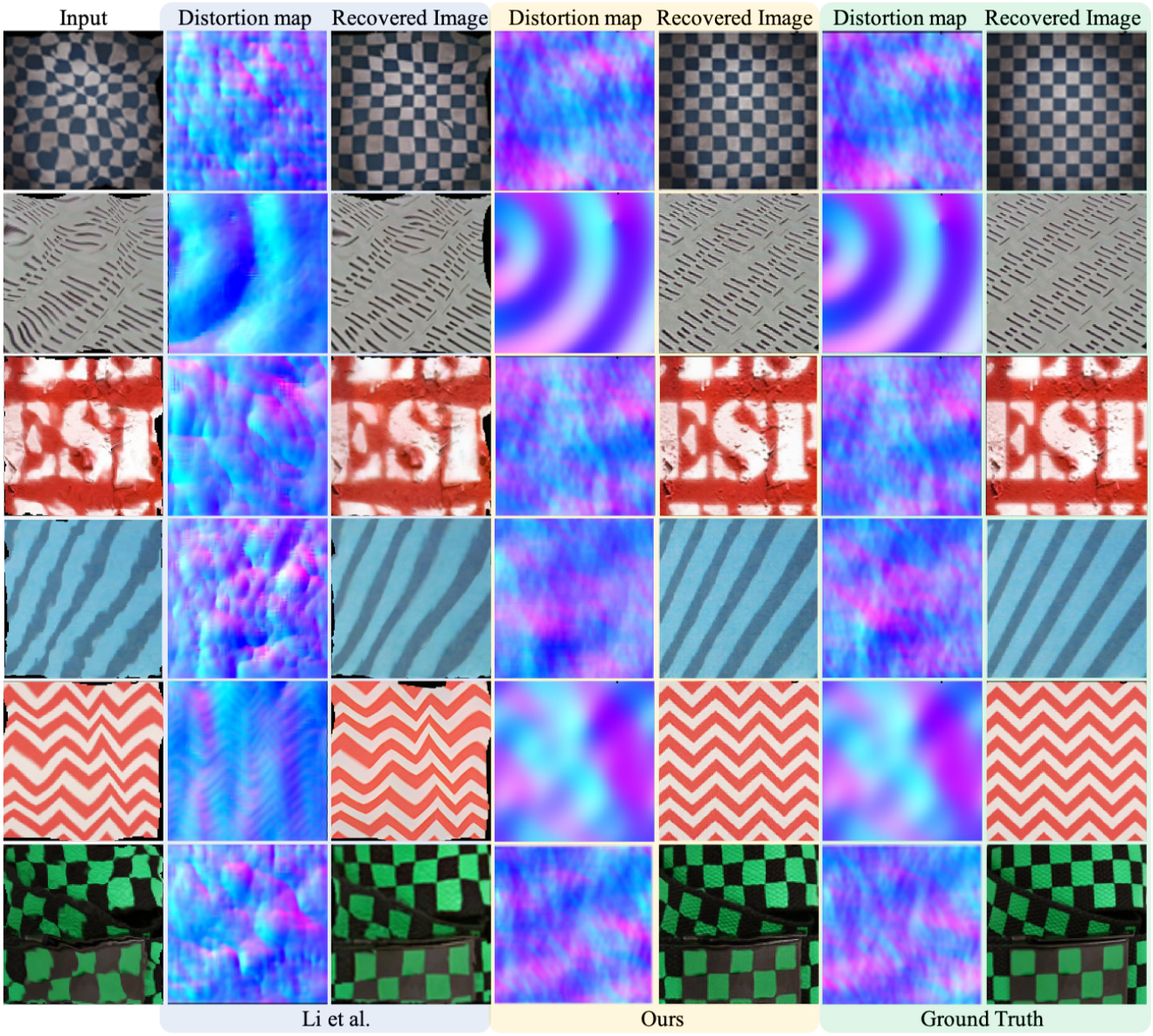

Figure 4. Visual comparison with [5] on the SynSet. We compare both the estimated distortion map and distortion-free image. We see that our predictions are robust to different types of wave fronts.

Figure 5. Visual comparison with the state-of-the-arts on the real captured TianSet *et al*. [9]. Here Tian-10, Oreifej-10, and James-10 refer to using the 10 frames of the "Middle", "Small", "Tiny" sequences from TianSet as input to the methods [9], [6], and [4], respectively.

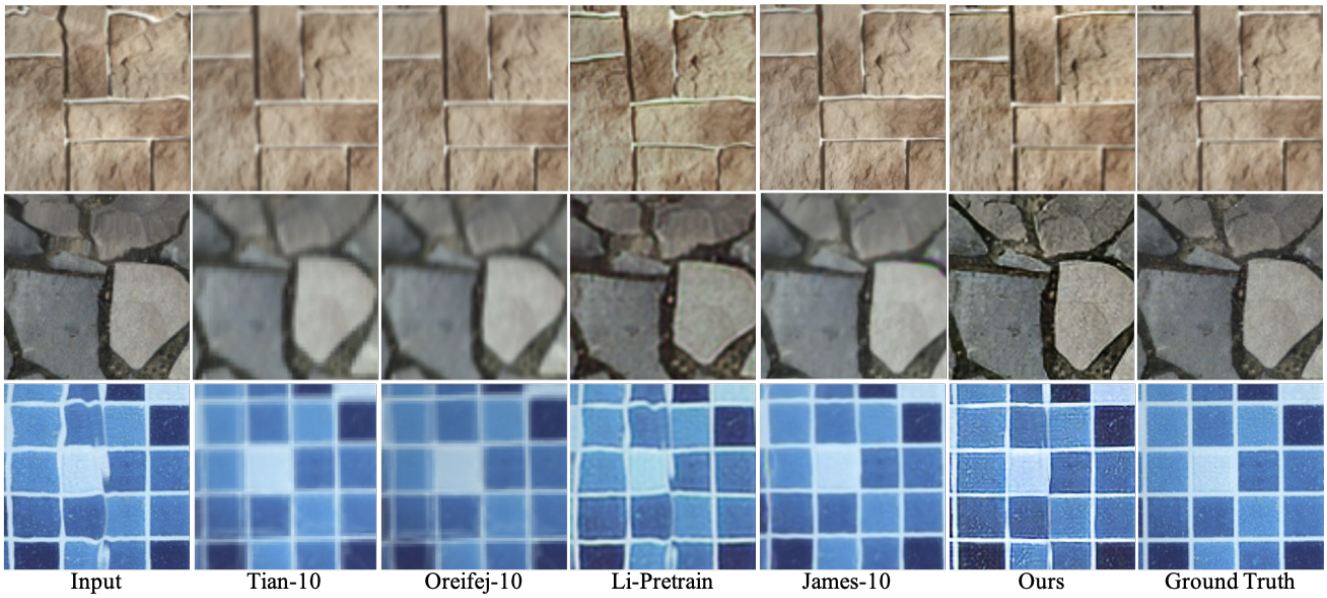

Figure 6. More visual comparative results on real underwater image dataset provided by Thapa *et al*. [8]. Here Tian-10, Oreifej-10, and James-10 refer to using the 10 frames of the three sequences from ThapaSet as input to the methods [9], [6], and [4], respectively.

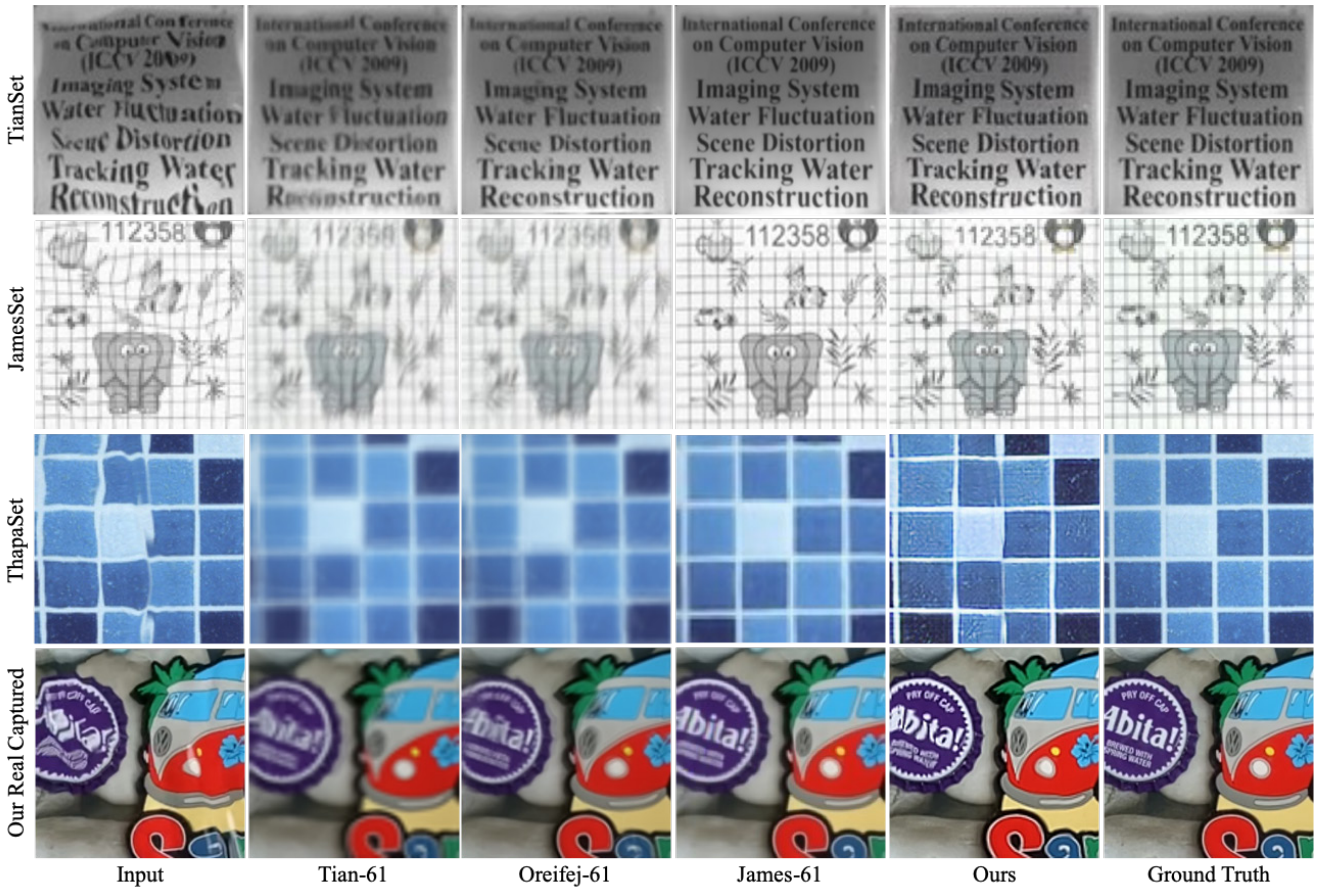

Figure 7. More visual comparative results on real underwater image dataset provided by Tian *et al*. [9], James *et al*. [4], Thapa *et al*. [8], and our real captured set. Note that all the comparision methods take 61 frames as input, i.e., Tian-61, Oreifej-61, and James-61 refer to using the 61 frames as inputs to the methods [9], [6], and [4], respectively. Our method takes only 3 images as its input.

3.4. Additional "In-The-Wild" Results

Since our method just requires three input frames, the images can easily be taken of a dynamic scene in either burst mode or a video at $\sim$ 60-120 fps. These settings are readily available in any modern day cell phone or drones. Fig. 8 is a real outdoor pool scene. It is captured by the lightweight DJI Mavic Mini drone. Fig. 9 is a real indoor aquarium setting where we see highly mobile fishes. It is captured with a cell phone camera. For more dynamic results, please refer to our `supplementary video`.

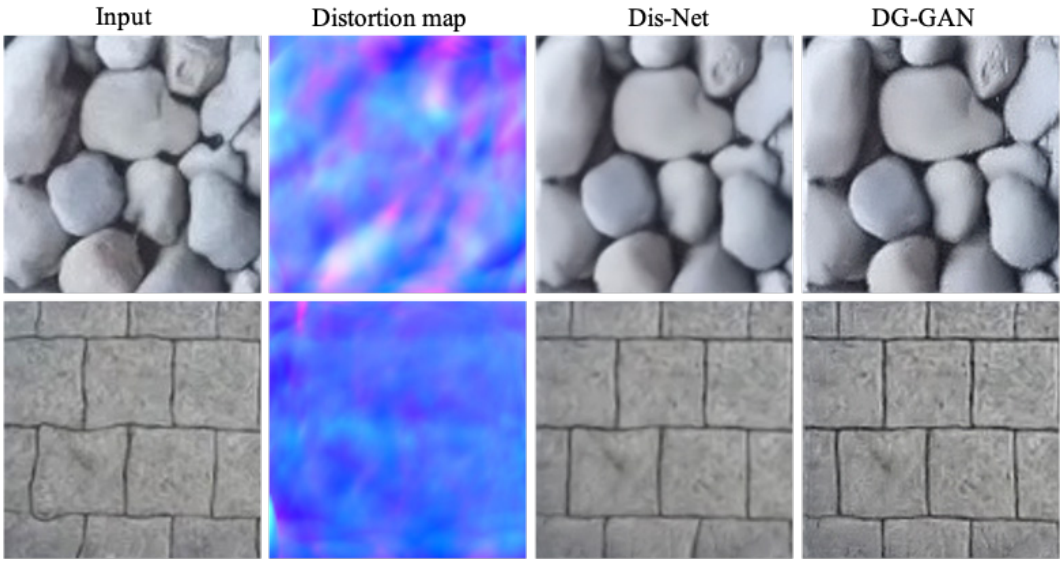

Figure 8. Visual results on Pool Scene taken with drone. The first column represents the distorted input image, the second column is our distortion-map estimation, the third column is the distortion-free image estimated by our Dis-Net and the last column is our final distortion-free image estimation from DG-GAN.

References

[1] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing textures in the wild. In *the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014. 2

[2] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *the International Conference on Pattern Recognition (ICPR)*, 2010. 4

[3] Quan Huynh-Thu and Mohammed Ghanbari. Scope of validity of psnr in image/video quality assessment. *Electronics letters*, 44(13):800–801, 2008. 4

[4] Jerin Geo James, Pranay Agrawal, and Ajit Rajwade. Restoration of non-rigidly distorted underwater images using a combination of compressive sensing and local polynomial image representations. In *the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 5, 7, 8

[5] Zhengqin Li, Zak Murez, David Kriegman, Ravi Ramamoorthi, and Manmohan Chandraker. Learning to see through turbulent water. In *the Winter Conference on Applications of Computer Vision (WACV)*, 2018. 5, 6

[6] Omar Oreifej, Guang Shu, Teresa Pace, and Mubarak Shah. A two-stage reconstruction approach for seeing through water. In *the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011. 4, 5, 7, 8

[7] Jerry Tessendorf. Simulating ocean water. *Simulating Nature: Realistic and Interactive Techniques. SIGGRAPH*, 1(2):5, 2001. 2, 3

[8] Simron Thapa, Nianyi Li, and Jinwei Ye. Dynamic fluid surface reconstruction unsing deep neural network. In *the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 5, 7, 8

[9] Yuandong Tian and Srinivasa G. Narasimhan. Seeing through water: Image restoration using model-based tracking. In *the Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009. 5, 7, 8

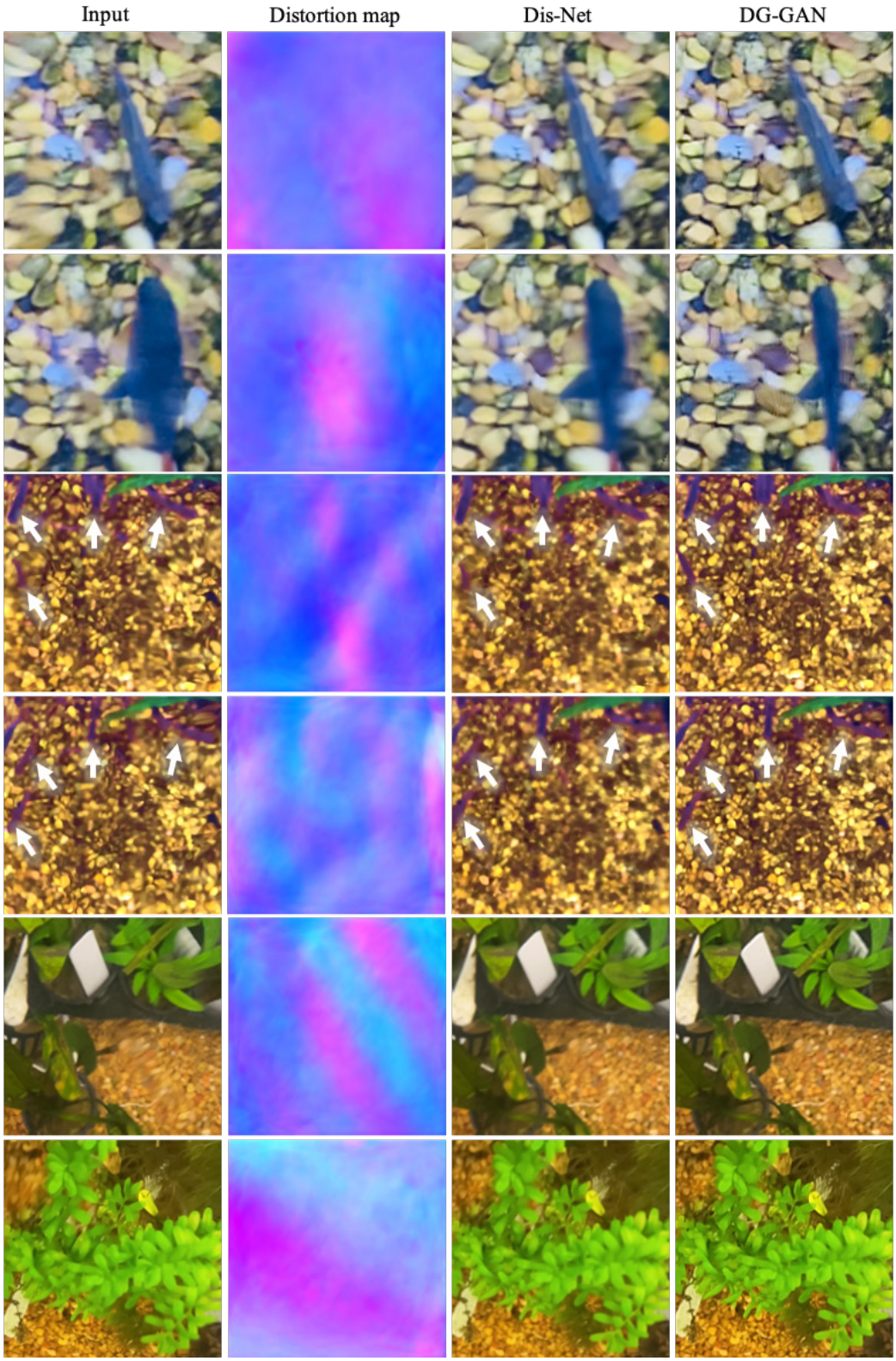

Figure 9. Visual results on aquarium scene with one big fish (top two rows) and multiple tiny fishes (middle two rows) swimming, and aquatic plants (bottom two rows). We use the white arrows to point to the tiny fishes. The first column represents the distorted input image, the second column is our distortion-map estimation, the third column is the distortion-free image estimated by our Dis-Net and the last column is our final distortion-free image estimation from DG-GAN.