# UGAN: Underwater Image Restoration using Generative Adversarial Networks

Cameron Fabbri[1], Md Jahidul Islam[2], and Junaed Sattar[3]

***Abstract*— Autonomous underwater vehicles (AUVs) rely on a variety of sensors – acoustic, inertial and visual – for intelligent decision making. Due to its non-intrusive, passive nature, and high information content, vision is an attractive sensing modality, particularly at shallower depths. However, factors such as light refraction and absorption, suspended particles in the water, and color distortion affect the quality of visual data, resulting in noisy and distorted images. AUVs that rely on visual sensing thus face difficult challenges, and consequently exhibit poor performance on vision-driven tasks. This paper proposes a method to improve the quality of visual underwater scenes using Generative Adversarial Networks (GANs), with the goal of improving input to vision-driven behaviors further down the autonomy pipeline. Furthermore, we show how recently proposed methods are able to generate a dataset for the purpose of such underwater image reconstruction[Reconstruction meaning what we are doing here, right? So this is a contribution too, to show how CycleGAN is used to generate the dataset, correct?]. For any visually-guided underwater robots, this improvement can result in increased safety and reliability through robust visual perception. To that effect, we present quantitative and qualitative data which demonstrates that images corrected through the proposed approach generate more visually appealing images, and also provide increased accuracy for an underwater diver tracking algorithm.**

## I. Introduction

TODO - talk about having to go back to the same location if you want to get good/bad pairs of the "same" image.**[I assume Cam will do this?]** Underwater robotics has been a steadily growing subfield of autonomous field robotics, assisted by the advent of novel platforms, sensors and propulsion mechanisms. While autonomous underwater vehicles are often equipped with a variety of sensors, visual sensing is an attractive option because of its non-intrusive, passive, and energy efficient nature. The monitoring of coral reefs [**?**], deep ocean exploration [**?**], and mapping of the seabed**[Citation?]** are a number of tasks where visually-guided AUVs and ROVs (Remotely Operated Vehicles) have seen widespread use. Use of these robots ensures humans are not exposed to the hazards of underwater exploration, as they no longer need to venture to the depths (which was how such tasks were carried out in the past). Despite the advantages of using vision, underwater environments pose unique challenges to visual sensing, as phenomena such as light refraction, absorption and scattering from suspended particles can greatly affect the optics of light. As an example, because red wavelengths are quickly absorbed by water, images tend to have a green or blue hue to them. As one goes deeper, this effect worsens, as more and more red hue is absorbed. This distortion is extremely non-linear in nature, and is affected by a large number of factors, such as the amount of light present (overcast versus sunny, operational depth), amount of particles in the water, time of day, and the camera being used. This may cause difficulty in tasks such as segmentation, tracking, or classification due to their indirect or direct use of color.

The authors are with the Department of Computer Science and Engineering, University of Minnesota-Twin Cities, MN 55455, USA.
{[1]fabbr013, [2]islam034, [3]junaed}@umn.edu

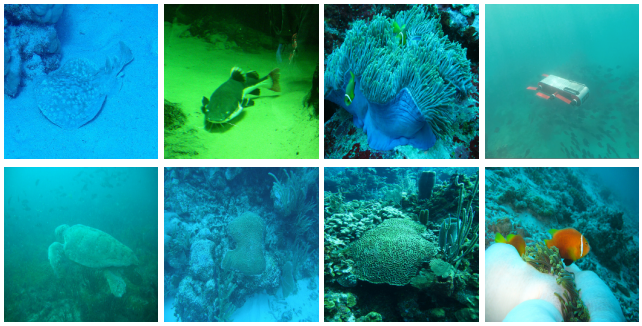


Fig. 1. Sample underwater images displaying the diversity of distortion that can occur. **Maybe say something about how some images lost all color, but some kept some color for close objects.**

As color and illumination begin to change with the depth, vision-based algorithms must need to be very generalizable in order to work within the depth ranges a robot may operate in. Because of the high cost and difficulty of acquiring a variety of underwater data to train a visual system on, as well as the high amount of noise introduced, algorithms may (and do) perform poorly in these different domains. Figure **??[DO NOT USE HARDCODED FIGURE NUMBERS!!! Use a label{} right before the caption, and then refer to it using the ref{} command as I have done here – see the LaTeXsource.]** shows the high variability in visual scenes that may occur in underwater environments. A step towards a solution to this issue is to be able to restore the images such that they appear to be above water, *i.e.*, with colors corrected and suspended particles removed from the scene. By performing a many-to-one mapping of these domains from underwater to not underwater (what the image would look like above water), algorithms that have difficulty performing across multiple forms of noise may be able to focus only one clean domain.

Deep neural networks have been shown to be powerful non-linear function approximators, especially in the field of vision**[Cite the canonical Deep Learning paper here please]**. Often times, these networks require large amounts of data, either labeled or paired with ground truth. For the

★ problem of automatically colorizing grayscale images**[CITE something on this?]**, paired training data is readily available due to the fact that any color image can be converted to black and white. However, underwater images distorted by either color or some other phenomenon lack ground truth, which is a major hindrance towards adopting a similar approach for correction. This paper proposes a technique based on Generative Adversarial Networks (GANs) to improve the quality of visual underwater scenes with the goal of improving the performance of vision-driven behaviors for autonomous

★ underwater robots**[Summarize the approach here. Then talk about datasets as you have done already.]** We use the recently proposed CycleGAN [**?**] approach, which learns to translate an image from any arbitrary domain $X$ to another arbitrary domain $Y$, as a way to generate a paired dataset.

★ By letting $X$ be a set of undistorted underwater images**[need to say where we find the undistorted ones]**, and $Y$ be a set of distorted underwater images, we can generate an image that appears to be underwater while retaining ground truth.

## II. Related Work

While there have been many very successful recent approaches towards automatic colorization [**?**], [**?**], most are focused on the task of grayscale to color. The work of [**?**] used an energy minimization formulation using a Markov Random Field. MORE RELATED BLAH. Many use physics based models to directly model the light source and such. Most similar to our line of work is the recently proposed WaterGAN [**?**], which uses GANs for color correction of underwater images. As ground truth pairs do not exist for underwater images, their first step is to structure a generator to create realistic underwater images. Their generator model can be broken down into three stages: 1) Attenuation, which accounts for range-dependent attenuation of light. 2) Scattering, which models the haze effect caused by photons scattering back towards the image sensor. 3) Vingetting, which produces a shading effect on the image corners that can be caused by certain camera lenses. Opposed to our work, they use a GAN for generating the underwater images and Euclidean loss for color correction, where as we use a GAN for both. Furthermore, they require depth information during the training of WaterGAN, whereas we only require two separate image domains.

Recent work in generative models, specifically GANs, have shown great success in areas such as inpainting [**?**], style transfer [**?**], and image-to-image translation [**?**], [**?**]. This is highly due to their ability to provide a more meaningful loss than simply the Euclidean distance, which has been shown to produce blurry results. In our work, we structure the problem as paired image-to-image translation, using Generative Adversarial Networks (GANs) as our generative model. Much like the work of [**?**], we use image pairs from two domains an input and ground truth.

Fig. 2. Paired samples of ground truth and distorted images generated by CycleGAN. Top row: Ground truth. Bottom row: Generated samples.

## III. Method

### A. Dataset Generation

Here we describe our dataset generation process. Depth, lighting conditions, camera model, and location are all factors that affect the amount of distortion an underwater image undergoes. Under certain conditions, it is possible that an underwater image may have very little distortion, or none at all. We let $I^C$ be an underwater image with no distortion, and $I^D$ be the same image with distortion. Our goal is to learn the function $f : I^D \rightarrow I^C$. Becasue of the difficulty of collecting underwater data, more often than not only $I^D$ or $I^C$ exist, but never both.

We use the recently proposed CycleGAN [**?**] to generate $I^D$ from $I^C$, which gives us a paired dataset of images. Given two datasets $X$ and $Y$, where $I^C \in X$ and $I^D \in Y$, CycleGAN learns a mapping $F : X \rightarrow Y$. Figure 2 shows paired samples generated from CycleGAN. From this paired dataset we train a generator $G$ to learn the function $f : I^D \rightarrow I^C$. It should be noted that during our data generation process, CycleGAN simultaneously learns a mapping $G : Y \rightarrow X$, which is similar to $f$. In section IV we show a comparison with our method.

### B. Adversarial Networks

GANs [**?**] are a class of generative models based on game theory in which a generator network competes against an adversary. Conditioned on an image $I^D$, the generator is trained to produce an  image to try and fool the discriminator, which is trained to distinguish be between distorted and non-distorted underwater images. In the original GAN formulation, the goal is to solve the minimax problem:

$$\min_G \max_D \mathbb{E}_{I^C \sim p_{train}(I^C)}[logD(I^C)] + \mathbb{E}_{I^D \sim p_{gen}(I^D)}[log(1 - D(G(I^D)))] \quad (1)$$

Note for simplicity in notation, we will further omit $I^C \sim p_{train}$ and $I^D \sim p_{gen}$. In this formulation, the discriminator is hypothesized as a classifier with a sigmoid cross entropy loss function, which in practice may lead to issues such as the vanish gradient and mode collapse. There have been many recent works which hypothesize a different loss function for the

discriminator [**?**], [**?**], [**?**], [**?**]. We focus on the Wasserstein GAN (WGAN) [**?**] formulation, which proposes to use the Earth-Mover or *Wasserstein-1* distance $W$ by constructing a value function using the Kantorovich-Rubinstein duality [**?**]. In this formulation, $W$ is approximated given a set of $k$-Lipschitz functions $f$ modeled as neural networks. To ensure $f$ is $k$-Lipschitz, the weights of the discriminator are clipped to some range $[-c, c]$. In our work, we adopt the Wasserstein GAN with gradient penalty (WGAN-GP) [**?**], which instead of clipping network weights like in [**?**], ensures the Lipschitz constraint by enforcing a soft constraint on the gradient norm of the discriminator's output with respect to its input. Following [**?**], our new objective then becomes

$$\mathcal{L}_{WGAN}(G, D) = \mathbb{E}[D(I^C)] - \mathbb{E}[D(G(I^D))] + \lambda_{GP}\mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}[(||\nabla_{\hat{x}}D(\hat{x})||_2 - 1)^2], \tag{2}$$

where $\mathbb{P}_{\hat{x}}$ is defined as samples along straight lines between pairs of points coming from the true data distribution and the generator distribution, and $\lambda_{GP}$ is a weighing factor. In order to give $G$ some sense of ground truth, as well as capture low level frequencies in the image, we also consider the $L1$ loss

$$\mathcal{L}_{L1} = \mathbb{E}[||I^C - G(I^D)||_1]. \tag{3}$$

Combining these, we get our final objective function for our network, which we call Underwater GAN (UGAN),

$$\mathcal{L}^*_{UGAN} = \min_G \max_D \mathcal{L}_{WGAN}(G, D) + \lambda_1 \mathcal{L}_{L1}(G). \tag{4}$$

### C. Image Gradient Difference Loss

Often times generative models produce blurry images. We explore a strategy to sharpen these predictions by directly penalizing the differences of image gradient predictions in the generator, as proposed by [**?**]. Given a ground truth image $I^C$, predicted image $I^P = G(I^C)$, and $\alpha \geq 1$, the Gradient Difference Loss (GDL) is given by

$$\mathcal{L}_{GDL}(I^C, I^P) = \sum_{i,j} ||I^C_{i,j} - I^C_{i-1,j}| - |I^P_{i,j} - I^P_{i-1,j}||^\alpha + ||I^C_{i,j-1} - I^C_{i,j}| - |I^P_{i,j-1} - I^P_{i,j}||^\alpha, \tag{5}$$

In our experiments, we denote our network as UGAN-P when considering the GDL, which can be expressed as

$$\mathcal{L}^*_{UGAN-P} = \min_G \max_D \mathcal{L}_{WGAN}(G, D) + \lambda_1 \mathcal{L}_{L1}(G) + \lambda_2 \mathcal{L}_{GDL} \tag{6}$$

### D. Network Architecture and Training Details

Our generator network is a fully convolutional encoder-decoder, similar to the work of [**?**], which is designed as a "U-Net" [**?**] due to the structural similarity between input and output. Encoder-decoder networks downsample (encode) the input via convolutions to a lower dimensional embedding, in which this embedding is then upsampled (decode) via transpose convolutions to reconstruct an image. The advantage of using a "U-Net" comes from explicitly preserving spatial dependencies produced by the encoder, as opposed to relying on the embedding to contain all of the information. This is done by the addition of "skip connections", which concatenate the activations produced from a convolution layer $i$ in the encoder to the input of a transpose convolution layer $n - i + 1$ in the decoder, where $n$ is the total number of layers in the network. Each convolutional layer in our generator uses kernel size $4 \times 4$ with stride 2. Convolutions in the encoder portion of the network are followed by batch normalization [**?**] and a leaky ReLU activation with slope $0.02$, while transpose convolutions in the decoder are followed by a ReLU activation [**?**] (no batch norm in the decoder). Exempt from this is the last layer of the decoder, which uses a TanH nonlinearity to match the input distribution of $[-1, 1]$.

Our discriminator is modeled after that of [**?**], except no batch normalization is used. This is due to the fact that we penalize the norm of the discriminator's gradient with respect to each input individually, which batch normalization would invalidate. The authors of [**?**] recommend layer normalization [**?**], but we found no significant improvements. In particular, the use of instance normalization [**?**] in the generator has been shown to improve quality, but we leave that for future work. In all of our experiments, we use $\lambda_1 = 100$, $\lambda_{GP} = 10$, batch size of 32, and the Adam Optimizer [**?**] with learning rate $1e-4$. Following WGAN-GP, the discriminator is updated $n$ times for every update of the generator, where $n = 5$. For UGAN-P, we set $\lambda_2 = 1.0$. Our implementation was done using the Tensorflow library [**?**]. [1] All networks were trained from scratch on a GTX 1080 for 100 epochs. Inference on the GPU takes on average $0.0138s$, which is about 72 Frames Per Second (FPS). On a CPU (Intel Core i7-5930K), inference takes on average $0.1244s$, which is about 8 FPS. We find both of these measures acceptable for underwater tasks.

## IV. Experiments

### A. Datasets

We used several subsets of Imagenet [**?**] for training and evaluation of our methods. We also evaluate a tracking algorithm on a diving video taken from Youtube [2]. Subsets of Imagenet containing underwater images were selected for the training of CycleGAN, and manually separated into two classes. We let $X$ be the set of underwater images with no distortion, and $Y$ be the set of underwater images with distortion. $X$ contained 6143 images, and $Y$ contained 1817 images. We then trained CycleGAN to learn the mapping $F : X \rightarrow Y$, such that images from $X$ appeared to have come from $Y$. Finally, our image pairs for training data were generated by distorting all images in $X$ with $F$. Figure 2 shows sample training pairs. When comparing with

[1] Code is available at https://github.com/cameronfabbri/Underwater-Color-Correction

[2] https://www.youtube.com/watch?v=QmRFmhILd5o

CycleGAN, we used a test set of 56 images acquired from Flickr [3].

### B. Evaluation

We train UGAN and UGAN-P on the image pairs generated by CycleGAN, and evaluate on the images from the test set, $Y$. Note that these images do not contain any ground truth, as they are original distorted images from Imagenet. Images for training and testing are of size $256 \times 256 \times 3$ and normalized between $[-1, 1]$. Figure 3 shows samples from the test set. Notably, these images contain varying amounts of noise. Both UGAN and UGAN-P are able to recover lost color information, as well as correct any color information this is present.

While many of the distorted images contain a blue or green hue over the entire image space, that is not always the case. In certain environments, it is possible that objects close to the camera are undistorted with correct colors, while the background of the image contains distortion. In these cases, we would like the network to only correct parts of the image that appear distorted. The last row in Figure 3 shows a sample of such an image. The orange of the clownfish is left unchanged while the distorted sea anemone in the background has its color corrected.

We use three metrics for a quantitative evaluation. First, we compare to CycleGAN, as it inherently learns an inverse mapping during the training of $F : X \rightarrow Y$, using the Canny edge detector [**?**] as it provides a color agnostic evaluation of the images in comparison to ground truth. Second, we compare local image patches to provide sharpness metrics on our images. Lastly, we show how an existing tracking algorithm for an underwater robot improves performance with generated images.

### C. Comparison to CycleGAN

It is important to note that during the process of learning a mapping $F : X \rightarrow Y$, CycleGAN also learns a mapping $G : Y \rightarrow X$. Here we show that UGAN and UGAN-P provide superior performance. We use a Canny edge detector [**?**] to provide a color agnostic evaluation of the images, as the original image contains distorted colors. Figure 4 shows the resulting images. We noticed that CycleGAN was mostly able to restore color, however the images contained various amounts noise. Because restoring color information should not alter the overall structure of the image, we measure the distance in the image space between the edges found in the original and the generated images. Table 1 provides the measurements from Figure 4, as well as the average over the entire dataset.

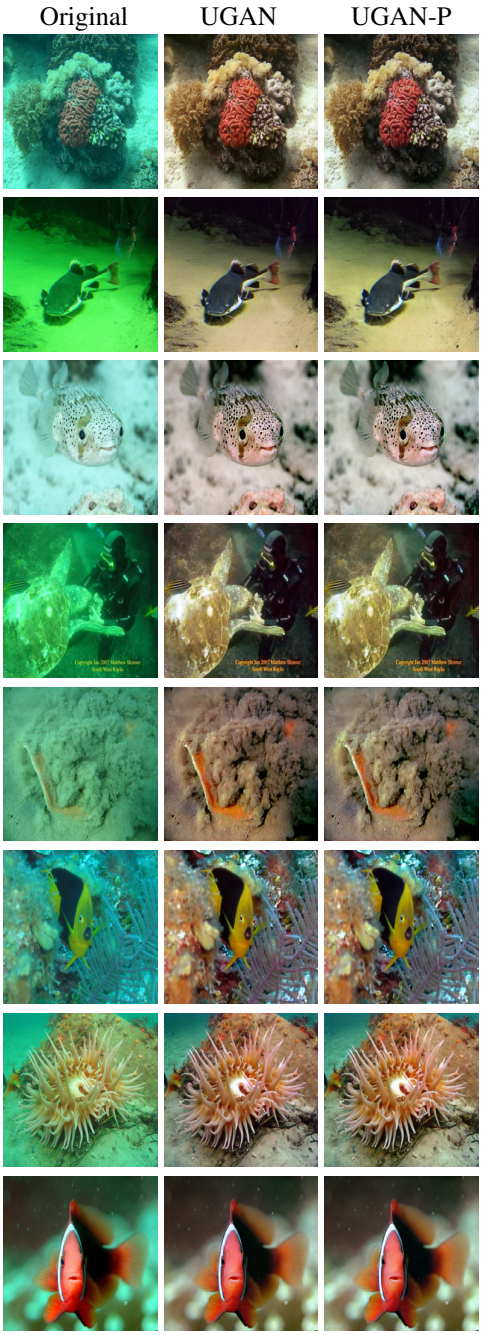



Fig. 3. Samples from our Imagenet testing set. The network can both recover color and also correct color if a small amount is present.

TABLE I

DISTANCES IN IMAGE SPACE

| Row/Method | CycleGAN | UGAN | UGAN-P |
|---|---|---|---|
| A | 116.45 | 85.71 | 86.15 |
| B | 114.49 | 97.92 | 101.01 |
| C | 120.84 | 96.53 | 97.57 |
| D | 129.27 | 108.90 | 110.50 |
| Mean | 111.60 | 94.91 | 96.51 |

[3]How do I cite Flickr

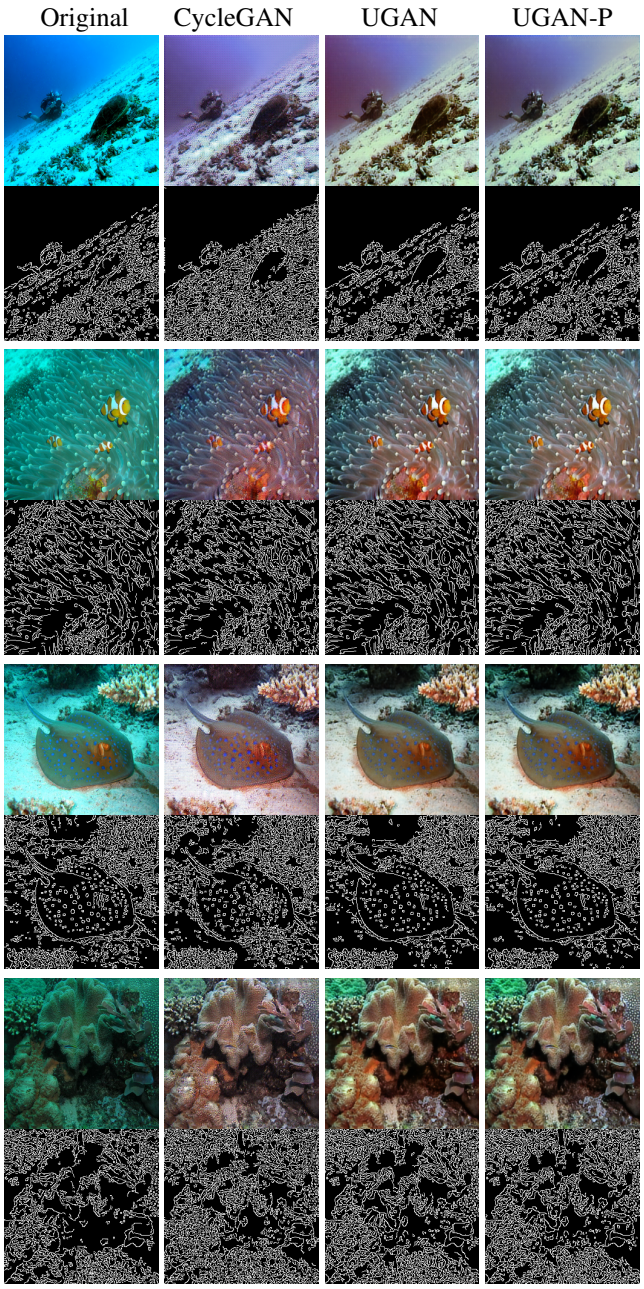



Fig. 4. Running the Canny Edge Detector on sample images. Both variants of UGAN contain less noise than CycleGAN, and are closer in the image space to the original. For each pair, the top row is the input image, and bottom row the result of the edge detector.

### *D. Diver Tracking using Frequency-Domain Detection*

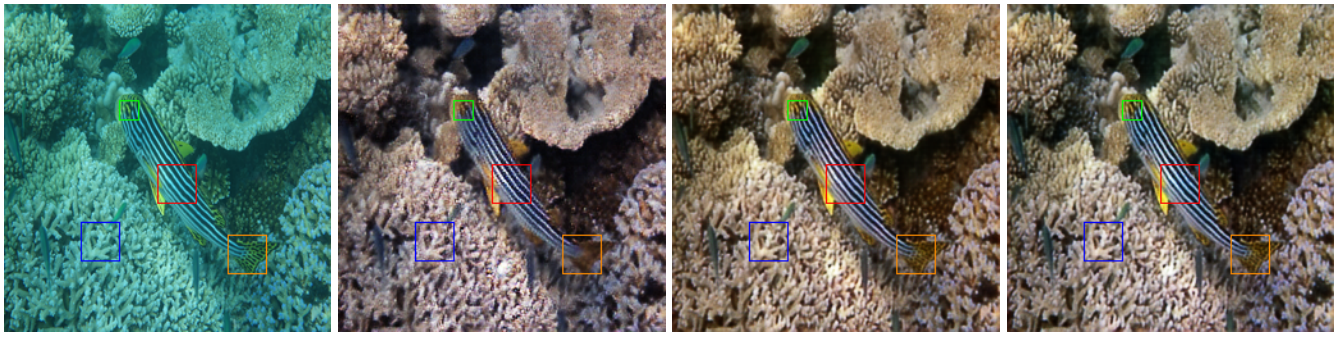
We investigate the frequency-domain characteristics of the restored images through a case-study of periodic motion tracking in sequence of images. Particularly, we compared the performance of Mixed Domain Periodic Motion (MDPM)- tracker [**?**] on a sequence of images of a diver swimming in arbitrary directions. MDPM tracker is designed for underwater robots to follow scuba divers by tracking distinct frequency-domain signatures (high-amplitude spectra at 1-2Hz) pertaining to human swimming. Amplitude spectra in frequency-domain correspond to the periodic intensity variations in image-space over time, which is often eroded in noisy underwater images [**?**].

Fig. **??** illustrates the improved performance of MDPM tracker on generated images compared to the real ones. Underwater images often fail to capture the true contrast in intensity values between foreground and background due to low visibility. The generated images seem to restore these eroded intensity variations to some extent, causing much improved positive detection for MDPM tracker.

## V. Conclusion

Future work: Getting a better and larger dataset. I think that would help a great deal. Also data augmentation after CycleGAN, such as gaussian blurring, noise, particle effect, more lighting effects etc.

## Acknowledgment



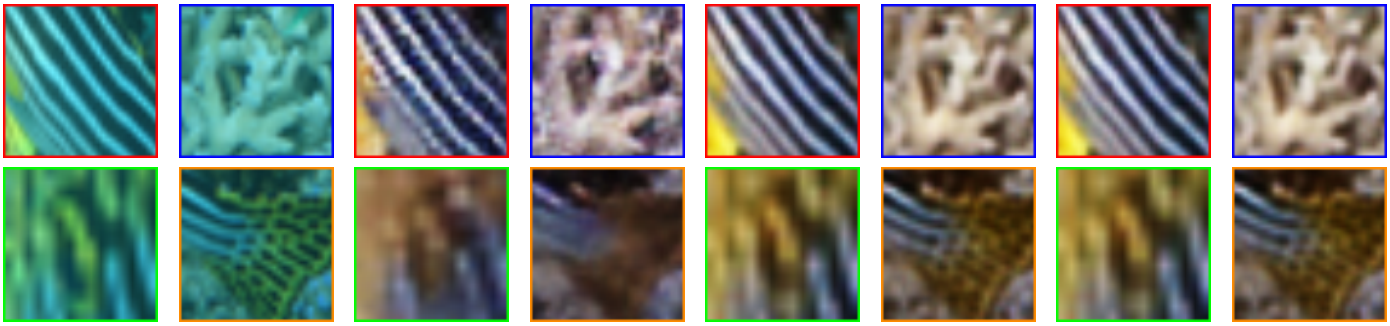
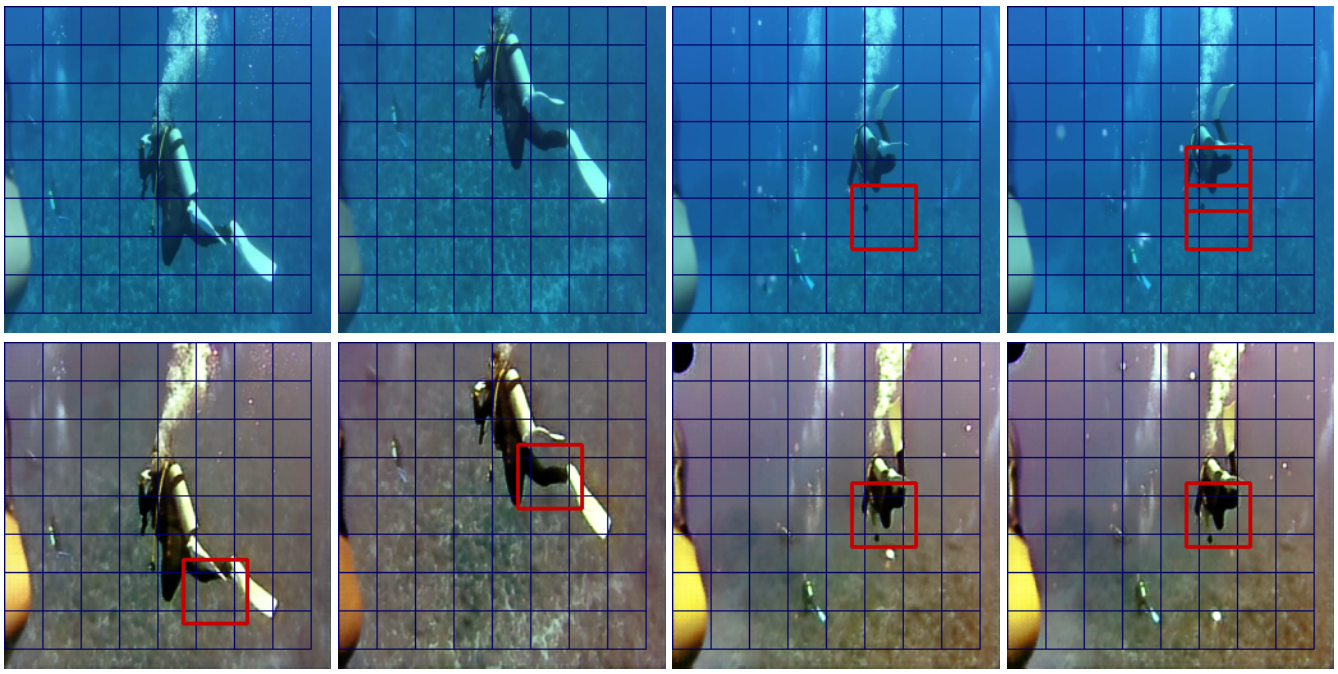


Fig. 5. Zooming in on comparisons.



| | Correct detection | Wrong detection | Missed detection | Total # of frames |
|---|---|---|---|---|
| On real images | 42 | 14 | 444 | 500 |
| On generated images | 147 | 24 | 329 | 500 |

Fig. 6. Performance of MDPM tracker [**?**] on both real (top row) and generated (second row) images; the Table compares the detection performance for both sets of images over a sequence of 500 frames.